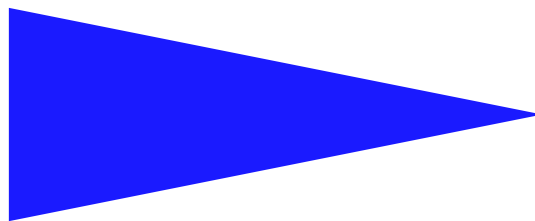


IRISA
INSTITUT DE RECHERCHE EN INFORMATIQUE ET SYSTÈMES ALÉATOIRES

PUBLICATION
INTERNE
N° 1864



CONTINUITY OF VARYING-FEATURE-SET CONTROL LAWS

NICOLAS MANSARD , ANTHONY REMAZEILLES AND
FRANÇOIS CHAUMETTE



CAMPUS UNIVERSITAIRE DE BEAULIEU - 35042 RENNES CEDEX - FRANCE

Continuity of Varying-Feature-Set Control Laws

Nicolas Mansard^{*}, Anthony Remazeilles^{**} and François Chaumette^{***}

Systèmes cognitifs
Projet Lagadic

Publication interne n° 1864 — September 2007 — 72 pages

Abstract:

Classical sensor-based control laws are based on the regulation of a set of sensor-based features to a desired reference value. The feature set is generally constant. In this article, we focus on the study of sensor-based control laws whose feature set varies during the servo. In that case, we first show that the classical control laws that use an iterative least-square minimization are discontinuous, and cannot be applied to real robots. We then show that these discontinuities are due to the pseudo-inverse operator, which is not continuous at matrix rank change. To solve this problem, we propose a new inversion operator. This operator is equal to the classical pseudo-inverse operator in the continuous cases, and ensures the continuity everywhere. This operator is then used to build a new control law. This general control scheme is applied to visual servoing, to ensure the continuity when some visual features leave the camera field of view. The experiments prove the interest and the validity of our approach.

Key-words: Sensor-based control - velocity control - continuity - linear algebra - pseudo inverse - visual servoing - visibility constraint

(Résumé : *tsvp*)

^{*} nicolas.mansard@irisa.fr

^{**} anthony.remazeilles@irisa.fr

^{***} francois.chaumette@irisa.fr

Tâches à dimension variable et continuité de la loi de commande

Résumé :

Les lois de commande référencée capteur classiques sont construites pour assurer la régulation d'un ensemble de primitives sensorielles à une valeur désirée. L'ensemble des primitives de référence est alors généralement constant. Dans cet article, nous nous intéressons aux lois de commande référencées par rapport à un ensemble variable de primitives. Dans ce cas, nous montrons tout d'abord que les lois de commande classiques utilisant une minimisation itérative aux moindres carrés occasionnent des discontinuités importantes dans la loi de commande, ce qui entraîne des pics d'accélération inacceptables sur un robot réel. Nous montrons alors que ces discontinuités sont dues à la non continuité de l'opérateur pseudo inverse lorsque le rang de la matrice à inverser change. Pour résoudre ces discontinuités, nous proposons un nouvel opérateur d'inversion matricielle, qui est identique à l'opérateur classique dans les cas continus, et qui garantit la continuité en tout point. Cet opérateur permet ensuite la construction d'une loi de commande originale, qui est ensuite mise en œuvre dans le cadre de l'asservissement visuel, pour assurer la continuité de la loi de commande lors de la sortie d'un nombre quelconque de points du champ de vision.

Mots clés : Commande référencée capteur - commande en vitesse - continuité - asservissement visuel

Contents

1	Introduction	5
2	State of the Art	6
2.1	Evidence of discontinuities	6
2.2	Robust visual servoing	7
2.3	Continuous visual servoing despite changes of visibility	8
2.4	Region reaching control	9
2.5	Qualitative servoing	9
2.6	Synthesis	10
3	Definitions	12
3.1	Varying-feature-set task	12
3.2	Input redundancy and decoupling	12
4	Varying-feature-set control scheme	15
4.1	Control laws based on a varying-feature-set task	15
4.2	Full approximation (27)	16
4.3	No approximation (28)	16
4.3.1	Non-redundant input signal	17
4.3.2	Redundant input signal	18
4.3.3	Fully-redundant input signal	19
4.3.4	Discussion	20
4.4	Using the damped least square inverse (29)	20
4.5	Partial approximations (30) and (31)	21
4.6	Conclusion	24
5	Building a new inverse operator	25
5.1	Formal definition	25
5.2	Construction of a continuous inverse operator	25
5.3	Continuous control law	27
6	Experimental results	28
6.1	Visual servoing implementation	28
6.2	First experiment: non-redundant task	29
6.3	Second experiment: oscillations with a non-redundant task	35
6.4	Third experiment: full-redundant task	38
6.5	Fourth experiment: redundant task	40
7	Conclusion	44

A	Classical control laws versus varying feature sets	45
A.1	First experiment: comparing $(\mathbf{HJ})^+$ and $(\mathbf{HJ})^\dagger$	45
A.2	Second experiment: non-redundant task ($\dim \mathbf{e} = 6$)	47
A.3	Third experiment: full-redundant task ($\dim \mathbf{e} = 8$)	51
A.4	Fourth experiment: redundant (not full redundant) input ($\dim \mathbf{e} = 12$)	55
A.5	Fifth experiment: decoupled features	59
B	Continuous inverse versus varying feature sets	61
B.1	First experiment: $\dim \mathbf{e} = 2$	61
B.2	Second experiment: $\dim \mathbf{e} = 6$	61
B.3	Third experiment: $\dim \mathbf{e} = 12$	65

1 Introduction

Robotic sensor-based tasks are generally defined by a set of features computed from the sensor output that should be regulated to a desired value. Various control laws have been proposed to regulate such sensor-based task to zero. For example, a generic approach to build continuous and stable control laws is proposed in [21]. Usually, the number and the type of features in the set is constant. The continuity of the control law and the stability of the system is then generally obtained outside some singularities that have to be avoided [19].

In this paper, we focus on sensor-based schemes whose feature set is not constant. Some works have already been proposed to consider servo schemes based on such a varying feature set. In [11], the features are removed from the set when they can not be computed anymore due to sensor-visibility lost. Similarly, the features detected as outliers are removed from the set in [6]. On the opposite, a feature can be removed when it is close enough from its desired value, in order to give more freedom to the robotic system [5, 20].

The properties of the obtained control scheme are generally studied on a case-by-case basis, in particular by using some hypotheses specific to the studied system. The results are thus difficult to generalize to other systems. In the following we propose to study these control laws as a generic control scheme called varying-feature-set control scheme. In particular, we will prove that the control laws computed directly from the varying feature set are not continuous in the general case. From this observation, a solution that ensures the continuity will be proposed.

Some classical control laws are firstly recalled and unified in Section 2. Based on this unification, the varying-feature-set control scheme is defined in Section 3. We then prove in Section 4 that the classical control laws coming from this definition are all unable to ensure the continuity everywhere. To remove these discontinuities, a new inverse operator is proposed in Section 5. Some experiments using visual servoing [12] are finally presented in Section 6, as an experimental comparison of the classical control laws and the proposed solution.

2 State of the Art

In this section, we quickly recall some classical control laws whose feature set is varying during the servo. When presenting these control laws, an effort is made to homogenize the different notations used by each author.

Let us consider an error function \mathbf{e} such that:

$$\dot{\mathbf{e}} = \mathbf{J}\dot{\mathbf{q}} \quad (1)$$

where \mathbf{q} is the robot articular position and $\mathbf{J} = \partial \mathbf{e} / \partial \mathbf{q}$ is the jacobian of \mathbf{e} . The jacobian \mathbf{J} is a $k \times n$ -matrix, where n is the number of degrees of freedom (DOF) of the robot ($n = \dim \mathbf{q}$) and k is the size of the error function ($k = \dim \mathbf{e}$). The rank of \mathbf{J} is noted m . In (1), it is implicitly supposed that $\mathbf{e}(\mathbf{q})$ is derivable everywhere and does not depend on the time variable but through the articular position. Based on (1), a classical controller can be obtained:

$$\dot{\mathbf{q}} = -\lambda \widehat{\mathbf{J}}^+ \mathbf{e} \quad (2)$$

where λ is a positive parameter used as a gain to tune the convergence velocity, \mathbf{A}^+ stands for the pseudo-inverse (or least-square inverse) of matrix \mathbf{A} [2], and $\widehat{\mathbf{A}}$ is an approximation of \mathbf{A} .

Control law (2) ensures an exponential decrease of each component of \mathbf{e} until regulation $\mathbf{e} = \mathbf{0}$ if $\mathbf{J}\widehat{\mathbf{J}}^+ = \mathbf{I}_m$ (since in that case $\dot{\mathbf{e}} = -\lambda \mathbf{J}\widehat{\mathbf{J}}^+ \mathbf{e} = -\lambda \mathbf{e}$). The global asymptotic stability can be obtained as soon as \mathbf{J} and $\widehat{\mathbf{J}}$ are full-rank (*i.e.* $k = m$) and $\mathbf{J}\widehat{\mathbf{J}}^+ > \mathbf{0}$ [21]. In all other cases (*i.e.* $k > m$), only the local asymptotic stability can generally be demonstrated. This classical control scheme has been widely used for sensor-based control [10, 22, 12, 17, 16].

2.1 Evidence of discontinuities

We now underline the discontinuities that can occur when modifying the feature set. Let us consider a task \mathbf{e}_1 , controlling n_1 of the n DOF of the robot. At time t , we increase the task by adding a term \mathbf{e}_2 , controlling n_2 DOF. The task is now $\begin{bmatrix} \mathbf{e}_1 \\ \mathbf{e}_2 \end{bmatrix}$. If the feature set is abruptly modified, is it possible to keep the control law continuity?

If $n_1 < n$, then, obviously, the DOF not controlled by \mathbf{e}_1 and that are now controlled by \mathbf{e}_2 are subject to a discontinuity (passing from a zero control input to a non zero control input).

If $n_1 = n$, the problem is the same, whatever the value of n_2 . Before time t , all the DOF of the robot are controlled only by \mathbf{e}_1 . After time t , a trade-off is realized by the pseudo inverse to fulfill at the same time \mathbf{e}_1 and \mathbf{e}_2 , which produces a discontinuity in the general case. For example, when realizing a visual servoing based on a multi-point target, the control law is different when considering a four-point target or a five-point target. Therefore, passing from four points to five points causes a discontinuity (an example of such a discontinuity is given in the experimental results in Section 6).

The discontinuity becomes obvious within the following basic experiment: let us consider a two-axes robot. Before time t , the task is \mathbf{e}_1 , whose jacobian is $\mathbf{J}_1 = \begin{bmatrix} 1 & 0 \end{bmatrix}$. The control law is thus:

$$\dot{\mathbf{q}} = -\lambda \mathbf{J}_1^+ \mathbf{e}_1 = \begin{bmatrix} -\lambda \mathbf{e}_1 \\ 0 \end{bmatrix} \quad (3)$$

In particular, the control input on the second axis of the robot is null. At time t , we add the component \mathbf{e}_2 , whose jacobian is $\mathbf{J}_2 = \begin{bmatrix} 0 & 1 \end{bmatrix}$. The jacobian of the full task is now $\mathbf{J} = \mathbf{I}_2$, the identity. The control law is thus:

$$\dot{\mathbf{q}} = -\lambda \mathbf{J}^+ \begin{bmatrix} \mathbf{e}_1 \\ \mathbf{e}_2 \end{bmatrix} = \begin{bmatrix} -\lambda \mathbf{e}_1 \\ -\lambda \mathbf{e}_2 \end{bmatrix} \quad (4)$$

In particular, if \mathbf{e}_2 is not null, the control input on the second axis is not null, which corresponds to a discontinuity in the control. A similar discontinuity appears when decreasing the size of the task vector.

It is thus not possible to directly modify the task vector while preserving the continuity of the control law. In order to avoid the discontinuity, special cares have to be taken to smooth the control law when modifying the size of the task.

In the following, we will present several works where the classical control scheme (2) has been modified to take into account the varying dimension of the task \mathbf{e} [6, 11, 20, 5]. In that case, a discontinuity in the control law can occur at each change of the task dimension, *i.e.* when one or several features appear or disappear. We will emphasize the solutions that have been proposed in the different articles to prevent this discontinuity to happen, and unify these works in a unique mathematical formulation.

2.2 Robust visual servoing

In [6], the problem of outliers in the input-feature set is addressed by associating to each feature a weight computed from the confidence that this feature is not an outlier. The proposed error function is $\mathbf{e}_q = \mathbf{H}\mathbf{e}$, where $\mathbf{H} = \mathbf{Diag}(h_1, \dots, h_k)$ is a weighting matrix used to remove the outliers from the feature set. The weights h_i are computed from a robust estimation algorithm. They vary from 1 when the robust estimation gives full confidence to 0 when the feature is doubtlessly an outlier. The control law is computed by analogy with (2):

$$\dot{\mathbf{q}} = -\lambda (\mathbf{H}\hat{\mathbf{J}})^+ \mathbf{H}\mathbf{e} \quad (5)$$

When the confidence in a feature decreases, it is smoothly removed from the control law by decreasing the value of the corresponding vector component $h_i e_i$. Simultaneously, it is also smoothly removed from the feature set by nullifying the corresponding line of the jacobian matrix $\mathbf{H}\mathbf{J}$.

When using such robust visual servo, it is usually supposed that the number of visual features is quite important (e.g. $k = 100$). It can then be easily verified that $(\mathbf{H}\hat{\mathbf{J}})^+ \mathbf{H}$ is always full rank, even if several weights h_i tend toward 0. We will prove in the following (in Section 4.3.3) that this hypothesis is sufficient to prove the continuity of the control law.

2.3 Continuous visual servoing despite changes of visibility

In [11], the authors directly address the problem of the control-law continuity when the number of features varies, in the particular case where the variation is due to visibility lost. When considering the visibility problem, it is sometimes difficult to ensure that all the features will stay inside the camera field of view without dedicating a specific DOF to the visibility constraint as done in [7, 15]. When a feature leaves the field of view, it has to be removed from the feature set, which thereof causes the control law to be discontinuous.

In [11], a weight is added so that a feature is smoothly removed when it leaves the field of view. The error vector can be written $\mathbf{e}_q = \mathbf{W}\mathbf{e}$ where $\mathbf{W} = \text{Diag}(w_1, \dots, w_k)$ is the weighting matrix used to smoothly remove a feature when it leaves the field of view and $\mathbf{e} = (\mathbf{s} - \mathbf{s}^*)$ is the error between current and desired point positions. The weight w_i is null when the feature leaves the field of view, and is equal to 1 when the feature is at the center of the field of view. A control law that regulates this error function is proposed in [11]:

$$\dot{\mathbf{q}} = -\lambda (\mathbf{C}\mathbf{W}\mathbf{J})^{-1} \mathbf{C}\mathbf{W}(\mathbf{s} - \mathbf{s}^*) \quad (6)$$

where the matrix \mathbf{J} is supposed to be full column rank (the task \mathbf{e} controls all the DOF of the robot), and \mathbf{C} is a combination matrix used to ensure that the considered error function $\mathbf{C}\mathbf{W}(\mathbf{s} - \mathbf{s}^*)$ respects the task function requirement [21]. In [11] \mathbf{C} is set to $(\mathbf{W}^* \hat{\mathbf{J}})^+$, where \mathbf{W}^* is the value of the weight matrix at the desired position (typically, $\mathbf{W}^* = \mathbf{I}_k$, the identity matrix), and $\hat{\mathbf{J}}$ is an approximation of the jacobian.

It is possible to get a relation very similar to (5) by considering that $\hat{\mathbf{J}} = \mathbf{J}$. Notice that this is not the choice made in [11], where $\hat{\mathbf{J}}$ is set to the jacobian computed at the desired position. Nevertheless, this choice manages to obtain a similar behavior, and the discontinuity problem due to a point loss remains present. The control law can thus be written:

$$\begin{aligned} \dot{\mathbf{q}} &= -\lambda (\mathbf{J}^+ \mathbf{W}\mathbf{J})^{-1} \mathbf{J}^+ \mathbf{W}(\mathbf{s} - \mathbf{s}^*) \\ &= -\lambda ((\mathbf{J}^\top \mathbf{J})^{-1} \mathbf{J}^\top \mathbf{W}\mathbf{J})^{-1} (\mathbf{J}^\top \mathbf{J})^{-1} \mathbf{J}^\top \mathbf{W}(\mathbf{s} - \mathbf{s}^*) \end{aligned} \quad (7)$$

since \mathbf{J} is full column rank and thus $\mathbf{J}^+ = (\mathbf{J}^\top \mathbf{J})^{-1} \mathbf{J}^\top$. Finally, when enough features are in the field of view (*i.e.* when $(\mathbf{J}^\top \mathbf{W}\mathbf{J})$ is full column rank), the first term $((\mathbf{J}^\top \mathbf{J})^{-1} \mathbf{J}^\top \mathbf{W}\mathbf{J})^{-1}$ can be developed:

$$\begin{aligned} \dot{\mathbf{q}} &= -\lambda (\mathbf{J}^\top \mathbf{W}\mathbf{J})^{-1} \mathbf{J}^\top \mathbf{J} (\mathbf{J}^\top \mathbf{J})^{-1} \mathbf{J}^\top \mathbf{W}(\mathbf{s} - \mathbf{s}^*) \\ &= -\lambda (\mathbf{J}^\top \mathbf{W}\mathbf{J})^{-1} \mathbf{J}^\top \mathbf{W}(\mathbf{s} - \mathbf{s}^*) \end{aligned} \quad (8)$$

In other words, this last equation means that it is equivalent in that case to chose $\mathbf{C} = \mathbf{J}^+$ or $\mathbf{C} = \mathbf{J}^\top$. Let $\mathbf{H} = \sqrt{\mathbf{W}}$ be the diagonal matrix whose coefficients are the square-roots of the weights w_i . The control law proposed in [11] can finally be written:

$$\dot{\mathbf{q}} = -\lambda (\mathbf{H}\mathbf{J})^+ \mathbf{H}(\mathbf{s} - \mathbf{s}^*) \quad (9)$$

The final form of the control law is identical to the one obtained from [6]. Indeed, this control law also manages to progressively inactivate some lines of the feature set and jacobian matrix when the corresponding visual features are getting out of the camera field of view.

2.4 Region reaching control

In [5] the main purpose is to bring the end-effector of the robot to a region instead of a point. The goal region is defined as the intersection of a set of simple regions such as circles, half-planes, etc. Each region is defined analytically by an inequality. The control aims at reducing the error of each inequality until they are all satisfied. When an inequality is respected, the corresponding part of the control is stopped.

This work has been realized out of the task function approach [21]. It is presented here by using this formalism as a matter of comparison with the other control schemes presented in this section.

Like in the previous schemes, the control is realized using the articular velocity of the robot $\dot{\mathbf{q}}$. The control point is the robot end-effector, noted \mathbf{X} . The convergence region is defined using a set of k inequalities:

$$\forall i = 1..k, \quad e_i(\mathbf{X}) \leq 0 \quad (10)$$

If the control law was developed to ensure $\mathbf{e} = \mathbf{0}$, the robot would be controlled toward the intersection of all the contours of the regions (10). In order to bring the robotic system inside this region, the task vector is defined to be $\mathbf{e}_{\mathbf{q}} = \mathbf{H}(\frac{1}{2}e_1^2, \dots, \frac{1}{2}e_k^2)$, where \mathbf{H} is a diagonal matrix whose components are null if the corresponding region has been reached, and 1 otherwise:

$$h_i = \begin{cases} 0 & \text{if } \mathbf{X} \text{ satisfies (10)} \\ 1 & \text{otherwise} \end{cases} \quad (11)$$

The authors of [5] proposed the following control law:

$$\dot{\mathbf{q}} = -\lambda \mathbf{J}^{\top} \mathbf{H} \mathbf{e} = -\lambda (\mathbf{H} \mathbf{J})^{\top} \mathbf{H} \mathbf{e} \quad (12)$$

since \mathbf{H} is idempotent and symmetric. Once more, the final form is very close to those obtained in (5) and (9). However, the control law is here computed using the transpose of the final jacobian matrix $(\mathbf{H} \mathbf{J})$, instead of the pseudo-inverse which is usually used within the task function formalism, as in (5) and (9). We can note that the transpose operator is always continuous, whereas the pseudo-inverse operator is discontinuous at rank changes, as it will be shown later. As a result, it is easy to prove the control law (12) to be continuous, even when the number of active features (*i.e.* the number of non-zero h_i) decreases. On the other hand, the control obtained using \mathbf{J}^{\top} is not optimal compared to the control obtained with the pseudo-inverse \mathbf{J}^+ . In particular, the decrease of the error is not exponential, since $\dot{\mathbf{e}} = -\lambda \mathbf{J} \mathbf{J}^{\top} \mathbf{H} \mathbf{e}$

2.5 Qualitative servoing

The main objective of the qualitative servo control proposed in [20] is to enlarge the convergence area, by explicitly requiring that the error \mathbf{e} converges toward a confident interval, instead of a particular desired value as it is usually performed. In this sense, the robotic

system is thus required to perform a *qualitative convergence* (by opposition to a quantitative convergence toward a specific position).

The error function \mathbf{e}_q is defined with respect to \mathbf{e} , such that $\mathbf{e}_q = \mathbf{H}(\mathbf{e} - \bar{\mathbf{e}})$, where $\bar{\mathbf{e}}$ is the limit of the convergence area: when a component of \mathbf{e} is below its corresponding threshold in $\bar{\mathbf{e}}$, the associated part of the control is inactivated. The activation matrix \mathbf{H} is defined by:

$$\begin{aligned} \mathbf{H} : \mathbb{R}^m &\rightarrow \mathbb{R}^{m \times m} \\ \mathbf{a} &\rightarrow \mathbf{H} = \text{Diag}(h(a_1), \dots, h(a_m)) \end{aligned} \quad (13)$$

where

$$h(a) = \begin{cases} 0 & \text{if } a \leq 0 \\ 1 & \text{if } a \geq \epsilon \\ \frac{1}{2} \left(1 + \tanh\left(\frac{1}{1-a/\epsilon} - \frac{\epsilon}{a}\right) \right) & \text{otherwise} \end{cases} \quad (14)$$

The activation function h is C^∞ everywhere (even at the junction points 0 and ϵ). The continuity of the obtained error \mathbf{e}_q is thus the same as the original error \mathbf{e} .

A control law that regulates the error \mathbf{e} into the confidence interval has been proposed in [20]:

$$\dot{\mathbf{q}} = -\lambda(\mathbf{HJ})^+ \mathbf{H}(\mathbf{e} - \bar{\mathbf{e}}) \quad (15)$$

Once more, we recognize the same form of the previous control laws.

2.6 Synthesis

Several control laws that deal with varying feature set have been presented in the previous subsections. All have been written using an equivalent framework, with similar notations, as a matter of comparison. A common control law equation can be noticed:

$$\dot{\mathbf{q}} = -\lambda(\mathbf{HJ})^\boxplus \mathbf{H}\mathbf{e}, \quad (16)$$

where \boxplus is a matrix operator (the pseudo-inverse or the transpose in the control laws presented above).

The control law is a composition of three parts: the matrix \mathbf{HJ} , the matrix operator \boxplus , and the vector $\mathbf{H}\mathbf{e}$. The diagonal activation matrix \mathbf{H} is used to smoothly remove or add features of \mathbf{e} and also to nullify the corresponding lines of the jacobian matrix. When \boxplus is the pseudo inverse, this second point is fundamental. Indeed, if the jacobian line is not nullified (*i.e.* if $\dot{\mathbf{q}} = -\lambda\mathbf{J}^+ \mathbf{H}\mathbf{e}$ is used), then the feature is taken into account into the least-square minimization, and the control law will try to minimize the motion of the inactivated feature, by imposing the velocity $\dot{e}_i = 0$ (which is a control in itself, and the result is very different to not constrain \dot{e}_i at all).

To ensure the continuity of the control law, the simplest solution is to ensure the continuity of each of the component. In particular, it is not sufficient to ensure the continuity of \mathbf{HJ} and $\mathbf{H}\mathbf{e}$ if the matrix operator \boxplus is not continuous. Thanks to the use of \mathbf{H} , both the

task vector and the jacobian matrix are continuous at feature activation or inactivation. To ensure the control law continuity, it is then enough to ensure the continuity of the matrix operator \boxplus . Indeed, the pseudo inverse operator is continuous when the rank of the matrix is constant. It is very important to notice that this is always the case in the three control laws presented above. This is a sufficient condition to ensure the continuity of the control law.

However, the pseudo inverse operator is not continuous at rank change. This means that these control laws are not continuous if the number of features decreases below a certain level (which is not considered in these three works). In [5], this case can happen. The continuity of the control law is then obtained by using the transpose operator instead of the pseudo-inverse operator (the transpose is always continuous, even at matrix-rank change). However, using the transpose can lead to a very non-optimal control, and the pseudo-inverse is very often a much more convenient solution.

In the following sections, the generic control law (16) will be proved to be continuous as long as the number of active features is sufficient¹. It will also be shown that strong discontinuities can appear when the number of active features is not sufficient. Based on this observation, we will build a new matrix operator \boxplus that is continuous in all cases, and acts like the pseudo inverse outside of its discontinuities. From this new operator, a new control law of a similar form will be proposed and proved to be continuous in all cases.

¹More precisely, we will prove that the control law is continuous if the number of active features is sufficient to ensure that the Jacobian matrix $\mathbf{H}\mathbf{J}$ is always full row rank before and after the activation

3 Definitions

In this section, we define all the notions that are required for the following study. We firstly propose a global definition to refer to any task similar to those presented in the previous section. Then we propose formal definitions to characterize some classical notions of the redundancy of a system with respect to a given task.

3.1 Varying-feature-set task

Definition 3.1 (Varying-feature-set task) *Let \mathbf{e} be any sensor-based feature vector which is called task in the following. Its jacobian is supposed to be of constant rank. The task \mathbf{e}_q is a varying-feature-set task based on \mathbf{e} if it respects:*

$$\mathbf{e}_q = \mathbf{H}\mathbf{e} \quad (17)$$

where \mathbf{H} is a diagonal matrix whose coefficients continuously vary within the interval $[0, 1]$.

Remark 3.1: The four control schemes (5), (9), (12) and (15) recalled in the previous section are based on a varying-feature-set task.

3.2 Input redundancy and decoupling

Definition 3.2 (Full rank matrix) *The matrix \mathbf{A} is full row rank (FRR) iff the number of its rows is equal to its rank. It is full column rank (FCR) iff the number of its columns is equal to its rank.*

Definition 3.3 (Non-redundant input) *Let \mathbf{e} be any sensor-based task. The task \mathbf{e} is said to be non redundant in input (or to have a non-redundant input) if its jacobian matrix is FRR.*

Definition 3.4 (Redundant input) *On the opposite, a task has a redundant input if its jacobian is not FRR.*

Remark 3.2: If the matrix \mathbf{J} is not FRR, one can separate its lines into a generator set \mathbf{J}_0 and a redundant set \mathbf{J}_1 that can be defined as a linear combination of \mathbf{J}_0 : $\mathbf{J}_1 = \chi\mathbf{J}_0$. The couple (\mathbf{J}_0, χ) is called *factorization*. Its formal definition is the following.

Definition 3.5 (Matrix factorization) *Let \mathbf{J} be a non FRR matrix. Let \mathbf{P} be a permutation matrix, \mathbf{J}_0 a FRR matrix and χ a matrix such that:*

$$\mathbf{J} = \mathbf{P} \begin{bmatrix} \mathbf{J}_0 \\ \chi\mathbf{J}_0 \end{bmatrix}, \quad (18)$$

then the set $(\mathbf{P}, \mathbf{J}_0, \chi)$ is called factorization of the matrix \mathbf{J} by \mathbf{J}_0 . \mathbf{J}_0 is the generator matrix of \mathbf{J} , and χ is the multiplier of the factorization. In order to simplify notations, the permutation \mathbf{P} will be often omitted. Thereby, a factorization of \mathbf{J} is noted (\mathbf{J}_0, χ) (\mathbf{P} is easily deduced from \mathbf{J}, \mathbf{J}_0 and χ).

Remark 3.3: If some columns of the multiplier χ are null, the factorization can be developed:

$$\mathbf{J} = \begin{bmatrix} \mathbf{J}_A \\ \mathbf{J}_B \\ \chi_B \mathbf{J}_B \end{bmatrix} \quad (19)$$

with $\mathbf{J}_0 = \begin{bmatrix} \mathbf{J}_A \\ \mathbf{J}_B \end{bmatrix}$ and $\chi = [\mathbf{0} \ \chi_B]$. Features corresponding to \mathbf{J}_B and $\chi_B \mathbf{J}_B$ are the redundant part of the input vector. \mathbf{J}_A corresponds to the non-redundant part, because none of the features can be defined as a linear combination of the features associated to \mathbf{J}_A . On the opposite, if the multiplier χ does not have any null column, then the factorization is said to be fully redundant.

Definition 3.6 (Full redundant input) *The task \mathbf{e} has a full redundant input if it is not possible to find any partition of its jacobian \mathbf{J} of the form (19) with $\mathbf{J}_A \neq \mathbf{0}$.*

Corollary 3.1 (Characterization of a full redundant input) *The task \mathbf{e} has a full redundant input iff its jacobian \mathbf{J} can be written $\mathbf{J} = \mathbf{P} \begin{bmatrix} \mathbf{J}_0 \\ \chi \mathbf{J}_0 \end{bmatrix}$, where \mathbf{P} is a permutation matrix, \mathbf{J}_0 is FRR and none of the columns of the multiplier χ is null.*

Proof: Suppose that a factorization $(\mathbf{P}_0, \mathbf{J}_0, \chi_0)$ of \mathbf{J} exists, where all columns of χ are not null. Let $(\mathbf{P}_A, \mathbf{J}_A, \chi_A)$ be any factorization of \mathbf{J} . We have:

$$\mathbf{J} = \mathbf{P}_0 \begin{bmatrix} \mathbf{J}_0 \\ \chi_0 \mathbf{J}_0 \end{bmatrix} = \mathbf{P}_A \begin{bmatrix} \mathbf{J}_A \\ \chi_A \mathbf{J}_A \end{bmatrix} = \mathbf{P}_0 \begin{bmatrix} \mathbf{P} \mathbf{J}_A \\ \mathbf{P} \chi_A \mathbf{P}^{-1} \mathbf{P} \mathbf{J}_A \end{bmatrix} \quad (20)$$

with $\mathbf{P} = \mathbf{P}_0^{-1} \mathbf{P}_A$. By identification, we get $\chi_0 = \mathbf{P} \chi_A \mathbf{P}^{-1}$. The matrix \mathbf{P} is a permutation, and thus does not have any null column. Furthermore, the multiplier χ_0 does not have any null column as well. Then χ_A does not have any null column. Finally, if there exists a totally redundant decomposition of \mathbf{J} , then all the other possible decompositions are totally redundant. The feature set is indeed totally redundant.

The other implication of the equivalence is immediate. \square

Definition 3.7 (Decoupled input feature) *Let \mathbf{e} be a feature set. The feature e_2 is distinguished from the other features noted \mathbf{e}_1 . Let \mathbf{J}_1 and \mathbf{J}_2 be the jacobians of \mathbf{e}_1 and e_2 respectively. The feature e_2 is said decoupled from the other features \mathbf{e}_1 of \mathbf{e} if:*

$$R(\mathbf{J}_1^+) \perp R(\mathbf{J}_2^+) \quad (21)$$

where $R(\mathbf{A})$ is the range of matrix \mathbf{A} .

Corollary 3.2 *If two features sets \mathbf{e}_1 and \mathbf{e}_2 are decoupled, then:*

$$\mathbf{J}_2 \mathbf{J}_1^+ = \mathbf{0} \quad (22a)$$

$$\mathbf{J}_1 \mathbf{J}_2^+ = \mathbf{0} \quad (22b)$$

The reciprocal implication is also valid.

Proof: A well-known result concerning the kernel and the range of a matrix \mathbf{A} is:

$$R(\mathbf{A}^\top) = N(\mathbf{A})^\perp \quad (23)$$

where $N(\mathbf{A})$ is the kernel of matrix \mathbf{A} , and \mathbf{E}^\perp is the orthogonal complementary of subspace \mathbf{E} . Using (21), it is possible to write $R(\mathbf{J}_1^\top) \subset N(\mathbf{J}_2)$. Since $R(\mathbf{J}_1^\top) = R(\mathbf{J}_1^+)$, this proves (22a).

Reciprocally, if (22a) is true, then $R(\mathbf{J}_1^\top) = R(\mathbf{J}_1^+) \subset N(\mathbf{J}_2)$. Since $R(\mathbf{J}_2^+) = R(\mathbf{J}_2^\top) = N(\mathbf{J}_2)^\perp$, then $R(\mathbf{J}_1^+)$ is orthogonal to $R(\mathbf{J}_2^+)$.

The dual equation is obtained by the same way. \square

Remark 3.4: If all the features of \mathbf{e} are decoupled, then it is easy to show that \mathbf{e} has a non-redundant input. Moreover, if one feature is decoupled, \mathbf{e} can not have a full-redundant input.

Intuitively, the behavior of the control law when progressively inactivating a feature will differ if a redundant feature, a non redundant or a decoupled one is considered. These four definitions can be enlarged to the varying-feature-set tasks.

Definition 3.8 (Characteristics of a varying-feature-set task) *Let \mathbf{e}_q be a varying feature set. The corresponding active task \mathbf{e}_A is constructed by considering only the input features whose weight in \mathbf{H} is not null. Furthermore:*

- *The varying-feature-set task \mathbf{e}_q has a non-redundant active input if the associate active task \mathbf{e}_A has non-redundant input.*
- *The varying-feature-set task \mathbf{e}_q has a redundant active input if the associate active task \mathbf{e}_A has a redundant input.*
- *The varying-feature-set task \mathbf{e}_q has a full-redundant active input if the associate active task \mathbf{e}_A has a full-redundant input.*
- *An active feature e_2 is decoupled from the other active features if e_2 is decoupled from the other features belonging to the associate active task \mathbf{e}_A .*

Using these definitions, we will now study the control laws based on varying feature set (such as those recalled in Section 2) in the general case.

4 Varying-feature-set control scheme

In this work, it is supposed that the robot is controlled in velocity. Within this context, a continuity break of the control law induces an infinite acceleration of the robotic system, which has to be avoided. Furthermore, a discontinuous control law may induce some instabilities. For example, if the sensor input slightly oscillates in the vicinity of the region provoking the discontinuity, then the noise will be amplified by the discontinuity, and the control law will be disturbed up to the point that no smoothing technique will be able to compensate the oscillation. The study of the continuity of the control law is thus an important point.

This section considers the continuity of different control laws derived from Definition 3.1, and presented in Section 4.1. It will be shown in Section 4.2 to 4.4 that when the jacobian of the task is not fully redundant, none of these control laws is continuous. Simulations given in Appendix A illustrate the revealed discontinuities.

4.1 Control laws based on a varying-feature-set task

Let \mathbf{e}_q be a task characterized by a varying-feature-set such that $\mathbf{e}_q = \mathbf{H}\mathbf{e}$. Its derivative is:

$$\dot{\mathbf{e}}_q = \mathbf{H}\dot{\mathbf{e}} + \dot{\mathbf{H}}\mathbf{e} \quad (24)$$

As seen in the previous section, it is usually considered that the weights of matrix \mathbf{H} are varying slowly. The time derivative of \mathbf{H} is thus usually approximated by $\mathbf{0}$. Let \mathbf{J} and $\mathbf{J}_{\mathbf{e}_q}$ be respectively the jacobians of \mathbf{e} and \mathbf{e}_q . With the approximation $\dot{\mathbf{H}} = \mathbf{0}$, a simple expression of the jacobian $\mathbf{J}_{\mathbf{e}_q}$ is obtained:

$$\mathbf{J}_{\mathbf{e}_q} = \mathbf{H}\mathbf{J} \quad (25)$$

Using the jacobian of the error \mathbf{e}_q , it is possible to write a control law that regulates \mathbf{e}_q , like in (2). If an exponential decoupled decreasing behavior is specified ($\dot{\mathbf{e}}_q = -\lambda\mathbf{e}_q$), the control law is obtained by inverting the jacobian matrix:

$$\dot{\mathbf{q}} = -\lambda\widehat{\mathbf{J}_{\mathbf{e}_q}^+}\mathbf{e}_q = -\lambda(\widehat{\mathbf{H}\mathbf{J}})^+\mathbf{H}\mathbf{e} \quad (26)$$

Several choices can be considered for $\widehat{\mathbf{J}_{\mathbf{e}_q}^+}$. Apart from the classical approximations of $\widehat{\mathbf{J}}$ (see [14] for a review), we will focus on five approximations concerning \mathbf{H} , in order to try to get the global continuity of the control law:

$$\dot{\mathbf{q}} = -\lambda(\widehat{\mathbf{H}\mathbf{J}})^+\widehat{\mathbf{H}}\mathbf{e} \quad (27)$$

$$\dot{\mathbf{q}} = -\lambda(\mathbf{H}\mathbf{J})^+\mathbf{H}\mathbf{e} \quad (28)$$

$$\dot{\mathbf{q}} = -\lambda(\mathbf{H}\mathbf{J})^\dagger\mathbf{H}\mathbf{e} \quad (29)$$

$$\dot{\mathbf{q}} = -\lambda(\widehat{\mathbf{H}}\mathbf{J})^+\mathbf{H}\mathbf{e} \quad (30)$$

$$\dot{\mathbf{q}} = -\lambda(\mathbf{H}\mathbf{J})^+\widehat{\mathbf{H}}\mathbf{e} \quad (31)$$

$\widehat{\mathbf{H}}$ is an approximation of \mathbf{H} defined as: $\widehat{\mathbf{H}} = \text{Diag}\left(\begin{cases} 1 & \text{if } h_i \neq 0 \\ 0 & \text{otherwise} \end{cases}\right)$, and \mathbf{A}^\dagger is the damped least square inverse of \mathbf{A} [18, 8] (the interest of this inverse operator will be given in the next paragraphs). It is trivial to obtain (28), (29) and (31) from the general relation (26). Since $\mathbf{H}^+\mathbf{H} = \widehat{\mathbf{H}}$, Eq. (27) and (31) are respectively obtained by approximating $\widehat{\mathbf{J}}_{\mathbf{e}_q}^+$ by $(\widehat{\mathbf{H}}\mathbf{J})^+\mathbf{H}^+$ and $(\mathbf{H}\mathbf{J})^+\mathbf{H}^+$. Even if these two derivations do not seem to be intuitive, their final formulations correspond to easily understandable situations, e.g. a full approximation and a partial one.

The following sections propose to study the behavior of these different control laws, and especially their continuity at jacobian rank change.

4.2 Full approximation (27)

This first control law corresponds to the naive way to consider a task with a varying feature set: a component getting inside the activation area is directly considered within the minimization scheme, without any progressive activation. Of course, this kind of control law is not continuous, as it has been shown in [11]. For example, let us consider a task \mathbf{e} that can be decomposed in $\mathbf{e} = (\mathbf{e}_1, e_2)$, where e_2 is of dimension 1. The corresponding jacobian can thus be written as: $\mathbf{J} = \begin{bmatrix} \mathbf{J}_1 \\ \mathbf{J}_2 \end{bmatrix}$. If one supposes that the feature e_2 passes from inactivation ($h_2 = 0$) to small non-null $h_2 > 0$ between iterations 0 and 1, then, at iteration 0, the control law (27) is:

$$\dot{\mathbf{q}} = -\lambda(\mathbf{H}\mathbf{J})^+\mathbf{H}\mathbf{e} = -\lambda\mathbf{J}_1^+\mathbf{e}_1, \quad (32)$$

whereas at instant 1, the control law is:

$$\dot{\mathbf{q}} = -\lambda \begin{bmatrix} \mathbf{J}_1 \\ \mathbf{J}_2 \end{bmatrix}^+ (\mathbf{e}_1, e_2) \quad (33)$$

The importance of the discontinuity depends on the value e_2 and also on the way the addition of the line \mathbf{J}_2 modifies the singular values of the jacobian.

4.3 No approximation (28)

To solve the discontinuity of (27), the logical solution is to use the smooth activation matrix \mathbf{H} . When a feature gets inside the activation area, it is thus progressively added within the control scheme, until full activation. The control law (28) can be found in [11, 6, 20], as presented in Section 2.

This section shows that these control laws are continuous as long as enough features are activated so that the input is redundant. It is also shown that the smoothness brought by \mathbf{H} is not effective when the number of active features is not sufficient. More precisely, it will be shown that control laws (28) and (27) are surprisingly equivalent when the task is non redundant. Three cases are separately studied, whether the task is not redundant, redundant or fully redundant.

4.3.1 Non-redundant input signal

Theorem 4.1 *Let \mathbf{e}_q be a varying-feature-set task whose active input is non-redundant. The two control laws (27) and (28) are equal.*

Proof: Let us first introduce another inverse of matrix \mathbf{A} , the generalized inverse [2]. It has been introduced in robotics in [23], and widely used since [3, 1]. A very good analysis of such an inverse can be found in [9]. Let \mathbf{W} be a full-rank square matrix. The weighted generalized inverse matrix of \mathbf{A} weighted on the left by the weights \mathbf{W} is defined to be [9]:

$$\mathbf{A}^{\mathbf{W}\#} = (\mathbf{W}\mathbf{A})^+ \mathbf{W} \quad (34)$$

The matrix $(\mathbf{H}\mathbf{J})^+ \mathbf{H}$ can be simply rewritten using the weighted inverse. Let us first define the weight matrix \mathbf{H}_f from \mathbf{H} :

$$\mathbf{H}_f = \text{Diag} \left(\begin{array}{cc} h_i & \text{if } h_i \neq 0 \\ 1 & \text{otherwise} \end{array} \right) \quad (35)$$

This matrix \mathbf{H}_f is full rank. Using this definition, we can write:

$$\begin{aligned} (\mathbf{H}\mathbf{J})^+ \mathbf{H} &= (\mathbf{H}_f \hat{\mathbf{H}}\mathbf{J})^+ \mathbf{H}_f \hat{\mathbf{H}} \\ &= (\hat{\mathbf{H}}\mathbf{J})^{\mathbf{H}_f\#} \hat{\mathbf{H}} \end{aligned} \quad (36)$$

since $\mathbf{H} = \mathbf{H}_f \hat{\mathbf{H}}$.

Let us now suppose that the first i features are activated (*i.e.* $\forall j = 1..i, h_j \neq 0$) while the $k - i$ last ones are disabled. This hypothesis can be easily done since it is just a matter of feature ordering. We suppose thus that the jacobian can be written:

$$\mathbf{H}\mathbf{J} = \begin{bmatrix} \mathbf{H}_1 \mathbf{J}_1 \\ \mathbf{0} \end{bmatrix} \quad (37)$$

where \mathbf{J}_1 is FRR since the active input of \mathbf{e}_q is supposed to be non-redundant.

One of the major results of [9] is to prove that the weighted inverse (34) is invariant to the choice of \mathbf{W} if \mathbf{A} is FRR. Since $\begin{bmatrix} \mathbf{A} \\ \mathbf{0} \end{bmatrix}^+ = [\mathbf{A}^+ \mathbf{0}]$, this result can easily be generalized to the case (37). Thus, since $\mathbf{H}_1 \mathbf{J}_1$ is FRR (\mathbf{J}_1 is FRR and \mathbf{H}_1 is invertible), it is possible to write using (36):

$$(\mathbf{H}\mathbf{J})^+ \mathbf{H} = (\hat{\mathbf{H}}\mathbf{J})^{\mathbf{H}_f\#} \hat{\mathbf{H}} = (\hat{\mathbf{H}}\mathbf{J})^+ \hat{\mathbf{H}} \quad (38)$$

This last result proves that if the active input features are non-redundant, the two control laws (27) and (28) are equal. \square

If the active input is not redundant, the weights of \mathbf{H} are not taken into account. The discontinuities are thus the same than when using the simple matrix $\hat{\mathbf{H}}$. This is experimentally verified in the comparison realized in Appendix A.2.

4.3.2 Redundant input signal

The previous result can be easily extended to the case of a redundant input as long as the input is not fully redundant.

Theorem 4.2 (Weighted least square invariance)

Let \mathbf{J} be any matrix, and \mathbf{W} be a diagonal and invertible weight matrix. Let \mathbf{J} be a factorization such that:

$$\mathbf{J} = \mathbf{P} \begin{bmatrix} \mathbf{J}_0 \\ \mathbf{J}_1 \\ \chi_1 \mathbf{J}_1 \end{bmatrix} \quad (39)$$

where $(\mathbf{J}_0, \mathbf{J}_1)$ is FRR. The same factorization can also be applied to the weighting matrix $\mathbf{W} = \mathbf{P} \text{Diag}(\mathbf{W}_0, \mathbf{W}_1, \mathbf{W}_2)$.

Then the non redundant part \mathbf{J}_0 of \mathbf{J} is invariant to the weights \mathbf{W} when computing the weighted general inverse $\mathbf{J}^{\mathbf{W}\#}$:

$$\mathbf{J}^{\mathbf{W}\#} = \mathbf{J}^{\widetilde{\mathbf{W}}\#} \quad (40)$$

where $\widetilde{\mathbf{W}} = \mathbf{P} \text{Diag}(\mathbf{I}, \mathbf{W}_1, \mathbf{W}_2)$.

Proof: We just have to prove that $\mathbf{J}^{\widetilde{\mathbf{W}}\#}$ is invariant to \mathbf{W}_0 .

Without any loss of generality, the permutation \mathbf{P} is supposed to be equal to \mathbf{I} (it is just a matter of feature ordering). Since $(\mathbf{J}_0, \mathbf{J}_1)$ is supposed FRR, \mathbf{J}_0 can be decomposed as:

$$\mathbf{J}_0 = \mathbf{J}_1^\perp + \zeta \mathbf{J}_1 \quad (41)$$

where \mathbf{J}_1^\perp belongs to the orthogonal of \mathbf{J}_1 . A simple full-rank decomposition of \mathbf{J} is then:

$$\mathbf{J} = \mathbf{F} \mathbf{G} = \begin{bmatrix} \mathbf{I} & \zeta \\ \mathbf{0} & \mathbf{I} \\ \mathbf{0} & \chi \end{bmatrix} \begin{bmatrix} \mathbf{J}_1^\perp \\ \mathbf{J}_1 \end{bmatrix} \quad (42)$$

where \mathbf{F} is FCR and \mathbf{G} is FRR. The pseudo inverse of \mathbf{J} weighted by \mathbf{W} can thus be decomposed through the full-rank decomposition:

$$(\mathbf{W}\mathbf{J})^+ \mathbf{W} = (\mathbf{W}\mathbf{F}\mathbf{G})^+ \mathbf{W} = \mathbf{G}^+ (\mathbf{W}\mathbf{F})^+ \mathbf{W} \quad (43)$$

Now, we just have to prove that $(\mathbf{W}\mathbf{F})^+ \mathbf{W}$ is invariant to the coefficients \mathbf{W}_0 corresponding to \mathbf{J}_0 . The matrix $(\mathbf{W}\mathbf{F})$ can be defined by:

$$(\mathbf{W}\mathbf{F}) = \begin{bmatrix} \mathbf{W}_0 & \mathbf{W}_0 \zeta \\ \mathbf{0} & \mathbf{W}_1 \\ \mathbf{0} & \mathbf{W}_2 \chi \end{bmatrix} \quad (44)$$

This matrix is block triangular. Its pseudo inverse can thus be analytically defined:

$$(\mathbf{W}\mathbf{F})^+ = \begin{bmatrix} \mathbf{W}_0^+ & \mathbf{A} \\ \mathbf{0} & \mathbf{B} \end{bmatrix} \quad (45)$$

with $\mathbf{B} = \begin{bmatrix} \mathbf{W}_1 \\ \mathbf{W}_2 \chi \end{bmatrix}^+$ and $\mathbf{A} = -\mathbf{W}_0^+ \mathbf{W}_0 \zeta \mathbf{B} = -\zeta \mathbf{B}$.

Finally, we get:

$$(\mathbf{W}\mathbf{F})^+ \mathbf{W} = \begin{bmatrix} \mathbf{I} & \mathbf{A}\mathbf{W}_{12} \\ \mathbf{0} & \mathbf{B}\mathbf{W}_{12} \end{bmatrix} \quad (46)$$

where $\mathbf{W}_{12} = \begin{bmatrix} \mathbf{W}_1 & \mathbf{0} \\ \mathbf{0} & \mathbf{W}_2 \end{bmatrix}$. Equation (46) is thus independent to the choice of \mathbf{W}_0 . \square

Corollary 4.1 *Let \mathbf{e}_q be a varying-feature-set task whose active input is redundant but not fully redundant. Then the non-zero weights h_i corresponding to the non-redundant features are not taken into account in the control law.*

Proof: By using the same notations than in the previous proof, the activation matrix can be written as $\mathbf{H} = \mathbf{H}_f \hat{\mathbf{H}}$, and $(\mathbf{H}\mathbf{J})^+ \mathbf{H}$ as $(\hat{\mathbf{H}}\mathbf{J})^{\mathbf{H}_f \#} \hat{\mathbf{H}}$. Theorem 4.2 specifies that this inverse is independent to the coefficients of \mathbf{H} corresponding to the non redundant part of \mathbf{J} . \square

Let us consider a task \mathbf{e}_q whose first feature is non redundant. The activation matrix is $\mathbf{H} = \begin{bmatrix} h & \mathbf{0} \\ \mathbf{0} & \mathbf{H}_1 \end{bmatrix}$. Using Corollary 4.1, we can write if h is not null:

$$(\mathbf{H}\mathbf{J})^+ \mathbf{H} = \left(\begin{bmatrix} 1 & \mathbf{0} \\ \mathbf{0} & \mathbf{H}_1 \end{bmatrix} \mathbf{J} \right)^+ \begin{bmatrix} 1 & \mathbf{0} \\ \mathbf{0} & \mathbf{H}_1 \end{bmatrix} \xrightarrow{h \rightarrow 0} \left(\begin{bmatrix} 1 & \mathbf{0} \\ \mathbf{0} & \mathbf{H}_1 \end{bmatrix} \mathbf{J} \right)^+ \begin{bmatrix} 1 & \mathbf{0} \\ \mathbf{0} & \mathbf{H}_1 \end{bmatrix} \quad (47)$$

However, if h is null:

$$(\mathbf{H}\mathbf{J})^+ \mathbf{H} = \left(\begin{bmatrix} 0 & \mathbf{0} \\ \mathbf{0} & \mathbf{H}_1 \end{bmatrix} \mathbf{J} \right)^+ \begin{bmatrix} 0 & \mathbf{0} \\ \mathbf{0} & \mathbf{H}_1 \end{bmatrix} \quad (48)$$

The matrix $(\mathbf{H}\mathbf{J})^+ \mathbf{H}$ is not continuous when $h \rightarrow 0$. The control law is thus not continuous when some features that are non redundant are inactivated.

4.3.3 Fully-redundant input signal

On the opposite, it is easy to show that (28) is continuous when the activated input is fully redundant. In this case, the activation or inactivation of any feature will not modify the rank of $\mathbf{H}\mathbf{J}$. The pseudo-inverse operator is continuous when the rank of the matrix is constant [2]. Since $\mathbf{H}\mathbf{J}$ is continuous, this proves that the control law is continuous. The experiments presented in Appendix A.3 illustrate the continuity obtained with this control law, whereas the ones with partial and full approximation present large peaks in the velocity sent to the controller.

In conclusion, the behavior of the controlled system is continuous when enough features are activated so that the input is fully redundant, and discontinuous when the input is only redundant or non-redundant. An experiment that sums up all these results is presented in Appendix A.4.

4.3.4 Discussion

The discontinuity of the classical solution (28) is due to the discontinuity of the pseudo-inverse operator when the rank of the jacobian matrix changes [2]. Numerically, this is easy to understand when looking to the singular value decomposition (SVD) of the pseudo-inverse. Let \mathbf{U} , $\mathbf{\Sigma}$, \mathbf{V} be the SVD of a matrix \mathbf{A} ($\mathbf{A} = \mathbf{U}\mathbf{\Sigma}\mathbf{V}^\top$), where \mathbf{U} and \mathbf{V} are orthonormal matrices, and $\mathbf{\Sigma}$ is a diagonal matrix composed of the singular values of \mathbf{A} . The pseudo inverse of \mathbf{A} is:

$$\mathbf{A}^+ = \mathbf{V}\mathbf{\Sigma}^+\mathbf{U}^\top \quad (49)$$

where $\mathbf{\Sigma}^+$ is a diagonal matrix whose each coefficient is the inverse of a singular value of \mathbf{A} when it is not null and 0 otherwise.

The rank of a matrix is directly linked to its SVD decomposition, through the number of non-zero singular values. When the matrix evolves continuously, a rank increase will thus correspond to the apparition of a non-zero singular value. Intuitively, the discontinuities in the pseudo-inverse operator are the same than those of the inverse function in zero. This explains intuitively that control law (28) is discontinuous when the rank of \mathbf{HJ} increases or decreases, even if the jacobian is continuous. When activating or inactivating a redundant feature, the rank of the jacobian does not change. This is always the case when considering a fully-redundant input task, which explains that (28) is always continuous when the input is fully redundant. However, at activation or inactivation of a non-redundant feature, the rank of \mathbf{HJ} increases or decreases. The discontinuities in the control law are then due to the discontinuities of the pseudo-inverse operator at rank change.

4.4 Using the damped least square inverse (29)

To compensate the discontinuities of the pseudo-inverse operator, it has been proposed to use the damped-least-square inverse [18, 13, 8]. Instead of using the inverse of the singular values when computing the pseudo-inverse, this solution uses another function that is equivalent to the inverse when the singular value is above a fixed threshold η and that does not tend to infinity when the singular value tends to zero. The obtained inverse of matrix \mathbf{A} is noted $\mathbf{A}^{\eta\dagger}$:

$$\mathbf{A}^{\eta\dagger} = \mathbf{V}\mathbf{\Sigma}^{\eta\dagger}\mathbf{U}^\top \quad (50)$$

where $\mathbf{\Sigma}^{\eta\dagger}$ is diagonal. Its coefficients $\sigma_i^{\eta\dagger}$ are computed from the singular values σ_i by:

$$\sigma_i^{\eta\dagger} = \frac{\sigma_i}{\sigma_i^2 + \eta^2} \quad (51)$$

Theoretically, the introduction of η does not bring any new parameter to tune, since it only replaces a threshold used when computing the pseudo-inverse to bound the singular value inversion. We thus note $\mathbf{A}^{\eta\dagger} = \mathbf{A}^\dagger$ in the following.

Using this operator, the obtained control law is (29) that we recall here:

$$\dot{\mathbf{q}} = -\lambda(\mathbf{HJ})^\dagger \mathbf{H}\mathbf{e}$$

In that case, the damping factor acts as a smoothing of the discontinuity, and we hope to obtain thereby the continuity of the control law. The smoothing is effective at rank changes (when a singular value passes from non-zero to zero), that is to say at activation or inactivation of a non-redundant feature. This corresponds to the discontinuities of (28). In theory, the introduction of the damping factor thus solves the problem of the discontinuity encountered in (28). However this is not the case in practice. Practically, the damping is only effective around the threshold η .

Let us consider the activation of a non-redundant feature from $\mathbf{H}_{h=0} = \begin{bmatrix} 0 & \mathbf{0} \\ \mathbf{0} & \mathbf{I} \end{bmatrix}$ to $\mathbf{H}_{h=1} = \mathbf{I}$. The matrix $(\mathbf{H}\mathbf{J})^\dagger \mathbf{H}$ is numerically equal to $(\mathbf{H}_0\mathbf{J})^\dagger \mathbf{H}_0$ when the corresponding singular value is very small compared to η , typically in $[0, 1e^{-3}\eta]$. It is numerically equal to $(\mathbf{H}_1\mathbf{J})^\dagger \mathbf{H}_1 = \mathbf{J}^+$ when the singular value is very large compared to η , typically in $[1e^{+3}\eta, 1]$. In practice, the matrix $(\mathbf{H}\mathbf{J})^\dagger \mathbf{H}$ passes from $(\mathbf{H}_0\mathbf{J})^\dagger \mathbf{H}_0$ to \mathbf{J}^+ in the very small interval $[1e^{-3}\eta, 1e^{+3}\eta]$. For example, if $\eta = 1e^{-6}$ as classically done, the smoothing is effective into the interval $[1e^{-9}, 1e^{-3}]$, that is to say on an interval whose length is only 10^{-3} , and the control law is discontinuous in practice.

On the opposite, when removing a redundant feature, the variation of $(\mathbf{H}\mathbf{J})^\dagger \mathbf{H}$ is smooth into $[0, 1]$, whose length is 1. The smoothing is effective on a large interval, and the control law is continuous in practice.

The comparison between the smoothness is shown on Fig. 1. As shown by Fig. 1-(a), the control law (29) is smooth in theory. However, as shown by Fig. 1-(b), both control laws (28) and (29) are equivalent in practice, and discontinuous.

4.5 Partial approximations (30) and (31)

We have shown in the previous section that the classical control laws (27) and (28) are unable to ensure the continuity of the control law when the input is not fully redundant. These discontinuities can be understood by writing that $(\mathbf{H}\mathbf{J})^+ \mathbf{H} = \mathbf{J}^+ \mathbf{H}^+ \mathbf{H} = \mathbf{J}^+$ (if \mathbf{H} is invertible and \mathbf{J} is FRR). The use of \mathbf{H} both inside and outside the pseudo-inverse operator induces intuitively a simplification that cancels the smoothing. To prevent this simplification, a logical proposition is to use the true activation matrix \mathbf{H} only once and its approximation $\hat{\mathbf{H}}$ elsewhere. Two possibilities arise then ; they correspond to (30) and (31).

Control law (30) smoothly nullifies the feature values that are getting close to the activation frontier, but abruptly removes the corresponding part of the jacobian matrix. The jacobian matrix is thus not continuous, but it is hoped to obtain a continuous control law by correcting the discontinuities with smoothed input features. The following theorem proves that the continuity is obtained only in the very particular case of a perfect decoupling.

Theorem 4.3 *Let \mathbf{e}_q be a varying-feature-set task based on \mathbf{e} . Control law (30) is continuous at feature activation or inactivation iff the activated features are decoupled from the other ones.*

Proof: Let us first consider a task \mathbf{e} where all the features but one are fully activated. This last feature is noted e_2 . Let \mathbf{J}_2 be its jacobian. The jacobian of the task \mathbf{e} can then

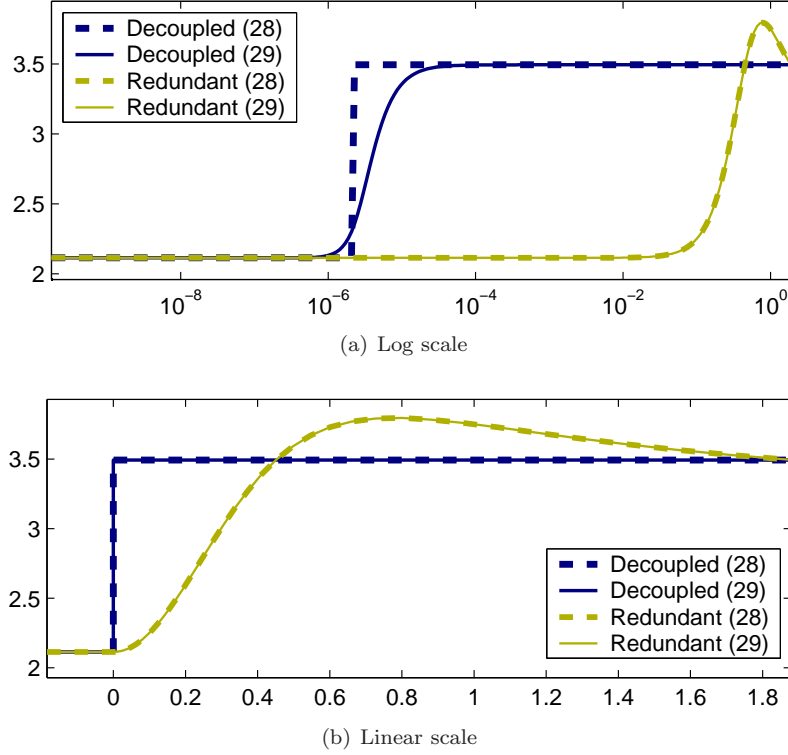


Figure 1: Comparison between the smoothness of $(\mathbf{HJ})^+\mathbf{H}$ and $(\mathbf{HJ})^\dagger\mathbf{H}$ when adding a non-redundant feature, and when adding a redundant feature. The graph is shown twice, with log X-scale (a), and with linear X-scale (b). Parameter η is set to $1e^{-6}$.

As shown in Section 4.3, control law (28) is discontinuous if the feature is non-redundant. The discontinuity occurs when the activation h is so small that the corresponding singular value is greater than η . On the opposite, control law (29) is continuous as shown on Fig. (a). However, the continuity occurs into a very small interval (here $[1e^{-7}, 1e^{-4}]$). In practice, the control law is very abrupt even if theoretically continuous. As shown by Fig. (b), the damping is practically inefficient to ensure the continuity. As a matter of comparison, the continuity is effective in a large interval (e.g. $[1e^{-3}, 1]$) when the feature is redundant. In that case, both (28) and (29) are numerically equal and the control law is smooth in practice.

be written $\mathbf{J} = \begin{bmatrix} \mathbf{J}_1 \\ \mathbf{J}_2 \end{bmatrix}$, where \mathbf{J}_2 is single-line. Since all the features corresponding to \mathbf{J}_1 are fully activated, the activation matrix can be written:

$$\mathbf{H} = \begin{bmatrix} \mathbf{I} & \mathbf{0} \\ \mathbf{0} & h \end{bmatrix} \quad (52)$$

The pseudo-inverse of \mathbf{J} can be decomposed using a *divide and conquer* approach:

$$\mathbf{J}^+ = \begin{bmatrix} \mathbf{J}_1 \\ \mathbf{J}_2 \end{bmatrix}^+ = \begin{bmatrix} \mathbf{J}_1^+ & \mathbf{J}_2^+ \end{bmatrix} + \mathbf{X}_{12} \quad (53)$$

where $[\mathbf{J}_1^+ \mathbf{J}_2^+]$ is the least-square minimization of each jacobian \mathbf{J}_1 and \mathbf{J}_2 taken separately, and \mathbf{X}_{12} is the part of the least-squares corresponding to the coupling between the two jacobians. To simplify the notations in the following, \mathbf{X}_{12} is written $\mathbf{X}_{12} = \begin{bmatrix} \mathbf{X}_{12}^1 & \mathbf{X}_{12}^2 \end{bmatrix}$, where \mathbf{X}_{12}^2 is simple-column.

Using this decomposition and (52), the inverse used in (30) can be written:

$$(\hat{\mathbf{H}}\mathbf{J})^+\mathbf{H} = \begin{cases} \begin{bmatrix} \mathbf{J}_1^+ & h\mathbf{J}_2^+ \end{bmatrix} + \begin{bmatrix} \mathbf{X}_{12}^1 & h\mathbf{X}_{12}^2 \end{bmatrix} & \text{if } h \neq 0 \\ \begin{bmatrix} \mathbf{J}_1^+ & \mathbf{0} \end{bmatrix} & \text{if } h = 0 \end{cases} \quad (54)$$

The discontinuity clearly appears in this formulation since:

$$\lim_{h \rightarrow 0} \left((\hat{\mathbf{H}}\mathbf{J})^+\mathbf{H} \right) = \begin{bmatrix} \mathbf{J}_1^+ & \mathbf{0} \end{bmatrix} + \begin{bmatrix} \mathbf{X}_{12}^1 & \mathbf{0} \end{bmatrix} \quad (55)$$

which is different from $\begin{bmatrix} \mathbf{J}_1^+ & \mathbf{0} \end{bmatrix}$ when \mathbf{X}_{12} is not null. To prove the equivalence between decoupling and continuity, we thus just have to prove that \mathbf{X}_{12} is null *iff* e_2 is decoupled. Let us suppose that $\mathbf{X}_{12} = \mathbf{0}$, that is to say

$$\mathbf{J}^+ = [\mathbf{J}_1^+ \mathbf{J}_2^+] \quad (56)$$

Due to the definition of the pseudo-inverse, it is possible to write $\mathbf{J}^+\mathbf{J}\mathbf{J}^+ = \mathbf{J}^+$. By introducing (56) in the former equation, we get the following development:

$$\mathbf{J}^+\mathbf{J}\mathbf{J}^+ = [\mathbf{J}_1^+\mathbf{J}_1\mathbf{J}_1^+ + \mathbf{J}_2^+\mathbf{J}_2\mathbf{J}_1^+ \quad \mathbf{J}_1^+\mathbf{J}_1\mathbf{J}_2^+ + \mathbf{J}_2^+\mathbf{J}_2\mathbf{J}_2^+] = \mathbf{J}^+ = [\mathbf{J}_1^+ \quad \mathbf{J}_2^+] \quad (57)$$

We thus obtain $\mathbf{J}_2^+\mathbf{J}_2\mathbf{J}_1^+ = \mathbf{0}$ and finally $\mathbf{J}_2\mathbf{J}_2^+\mathbf{J}_2\mathbf{J}_1^+ = \mathbf{J}_2\mathbf{J}_1^+ = \mathbf{0}$ (by multiplying both sides of the equality by \mathbf{J}_2). We obtain similarly $\mathbf{J}_1\mathbf{J}_2^+ = \mathbf{0}$. Using Corollary 3.2, this proves that e_1 and e_2 are decoupled. Reciprocally, suppose that e_2 is decoupled. Then $\mathbf{J}^+ = (\mathbf{J}_1^+\mathbf{J}_1 + \mathbf{J}_2^+\mathbf{J}_2)[\mathbf{J}_1^+ \mathbf{J}_2^+]^+$ [2]. By developing the previous equation, and introducing (22), we finally obtain $\mathbf{J}^+ = [\mathbf{J}_1^+ \mathbf{J}_2^+]$ and thus $\mathbf{X}_{12} = \mathbf{0}$. \square

	CONTROL LAW	NON REDUNDANT CASE		REDUNDANT CASE	
		Non redundant	Decoupled	Redundant	Fully redundant
(27)	$(\hat{\mathbf{H}}\mathbf{J})^+\hat{\mathbf{H}}$	discontinuous	discontinuous	discontinuous	discontinuous
(28)	$(\mathbf{H}\mathbf{J})^+\mathbf{H}$	discontinuous	discontinuous	discontinuous	continuous
(29)	$(\mathbf{H}\mathbf{J})^\dagger\mathbf{H}$	discontinuous	discontinuous	discontinuous	continuous
(30)	$(\hat{\mathbf{H}}\mathbf{J})^+\mathbf{H}$	discontinuous	continuous	discontinuous	discontinuous
(31)	$(\mathbf{H}\mathbf{J})^+\hat{\mathbf{H}}$	ill conditioned			

Figure 2: Summary of the control law behaviors (see section 2)

Theorem 4.3 is experimentally verified in the experiment presented in Appendix A.5.

In fact, (30) manages to smooth the part of the control corresponding to the minimization of e_2 . However, it is unable to smooth the part \mathbf{X}_{12}^1 of the control that corresponds to the coupling between e_2 and the other features. When h is not null, the space corresponding to the coupling matrix \mathbf{X}_{12}^1 is used as a trade-off to realize both \mathbf{e}_1 and e_2 . As soon as h really becomes null, \mathbf{X}_{12}^1 becomes instantaneously available for \mathbf{e}_1 alone, which results in a strong discontinuity in the control law.

Finally, Control Law (31) is not realizable due to ill-conditioning. Indeed, when some components of \mathbf{H} are small, the matrix $\mathbf{H}\mathbf{J}$ can be very ill-conditioned. Its pseudo-inverse $(\mathbf{H}\mathbf{J})^+$ is thus very large. In (28), the large coefficients of $(\mathbf{H}\mathbf{J})^+$ are diminished when multiplied with \mathbf{H} . If the approximation $\hat{\mathbf{H}}$ is used instead, no correction is brought since the small coefficients of \mathbf{H} are approximated by 1. The matrix coefficients of $(\mathbf{H}\mathbf{J})^+\hat{\mathbf{H}}$ can thus be very large, and the result on the robot control is unpredictable.

4.6 Conclusion

In this section, the definition of a varying-feature-set task has been given, and several control laws have been proposed, based on the classical methods in the state of art. The continuity of these control laws has been investigated when the number of visual features varies, and the general characteristics observed are summarized on Table 2. None of these control laws is continuous when the task is not fully redundant, *i.e.* when at least one feature can not be expressed as a combination of the others. Indeed, when this feature is activated or inactivated, the system earns or loses a degree of freedom. In this situation, the nice properties of continuity or stability demonstrated in [21] can not be obtained directly anymore.

5 Building a new inverse operator

Since the classical control schemes based on the pseudo-inverse are not able to ensure the continuity, a new control law, based on a new inversion operator is proposed in this section. We first propose a formal specification of the continuity properties that this operator should respect to ensure the control-law continuity. We propose then an implementation that fulfills these specifications, and we use it to build a new control law that is proved to be continuous.

5.1 Formal definition

In a first time, let us properly define the properties of the operator we are looking for. This operator should be equivalent to the classical pseudo-inverse operator when all the features are fully active or inactive (*i.e.* when $\forall i = 1..k, h_i \in \{0, 1\}$). The operator should also maintain the continuity when h goes smoothly from 0 to 1. This is formalized through the following definition:

Definition 5.1 (Continuous inverse) *Let \mathbf{A} be a matrix of size $(k \times n)$ and \mathbf{H} be a diagonal activation matrix of size $(k \times k)$, whose components belong to $[0, 1]$. The continuous inverse $\mathbf{A}^{\dagger \mathbf{H}}$ of a matrix \mathbf{A} subject to an activation \mathbf{H} respects the two following properties:*

- if $\forall i = 1..k, h_i \in \{0, 1\}$, then:

$$\mathbf{A}^{\dagger \mathbf{H}} = (\mathbf{H}\mathbf{A})^+ = (\mathbf{H}\mathbf{A})^+ \mathbf{H} \quad (58)$$

- The function $(\mathbf{A}, \mathbf{H}) \rightarrow \mathbf{A}^{\dagger \mathbf{H}}$ is continuous with respect to the variations of \mathbf{H} .

Remark 5.1: From this definition, it is clear that the continuous inverse ensures the continuity at full activation ($h \rightarrow 1$) and at full inactivation ($h \rightarrow 0$):

$$\mathbf{J}^{\dagger \text{Diag}(\mathbf{I}, h)} \xrightarrow{h \rightarrow 1} \mathbf{J}^+ \quad \text{and} \quad \mathbf{J}^{\dagger \text{Diag}(\mathbf{I}, h)} \xrightarrow{h \rightarrow 0} \begin{bmatrix} \mathbf{J}_1^+ & \mathbf{0} \end{bmatrix}$$

5.2 Construction of a continuous inverse operator

We now propose an implementation of this definition, based on the study of (30), and particularly the discontinuity observed in (55). The goal is to build an inverse of the following form:

$$\mathbf{J}^{\oplus \mathbf{H}} = \begin{bmatrix} h_1 \mathbf{J}_1^+ & h_2 \mathbf{J}_2^+ \end{bmatrix} + h_1 h_2 \mathbf{X}_{12} \quad (59)$$

The generalization of such relation requires a more formal definition of the coupling matrices.

Definition 5.2 (Coupling matrices of a matrix \mathbf{J}) *The coupling matrices of a $k \times n$ -matrix \mathbf{J} are indexed by the subspaces \mathcal{P} of the k first integers. They are defined recursively:*

$$\begin{aligned} & \text{if } \mathcal{P} = \emptyset, & \mathbf{X}_{\emptyset} &= \mathbf{0}_{n \times k} \\ & \text{otherwise } \forall \mathcal{P} \in \mathfrak{P}(k), & \mathbf{X}_{\mathcal{P}} &= \mathbf{J}_{\mathcal{P}}^+ - \sum_{\mathcal{Q} \subsetneq \mathcal{P}} \mathbf{X}_{\mathcal{Q}} \end{aligned} \quad (60)$$

where $\mathfrak{P}(k) = \mathfrak{P}([1..k]) = \left\{ \mathcal{P} \mid \mathcal{P} \subset [1..k] \right\}$ are all the subsets composed of the k first integers, and $\mathbf{J}_{\mathcal{P}} = \mathbf{H}\mathbf{J}$ where h_i is equal to 1 if $i \in \mathcal{P}$, and to 0 otherwise (i.e. $\mathbf{J}_{\mathcal{P}}$ is the jacobian matrix \mathbf{J} whose only activated lines are those of \mathcal{P}).

Remark 5.2:

- If $\mathcal{P} = \{i\}$, $i \in [1..k]$, then $\mathbf{X}_{\mathcal{P}} = \mathbf{X}_{\{i\}} = \begin{bmatrix} \mathbf{0}_{n \times (i-1)} & \mathbf{J}_i^+ & \mathbf{0}_{n \times (k-i)} \end{bmatrix}$, where \mathbf{J}_i is the i^{th} line of \mathbf{J} .
- If \mathbf{J} has two lines ($\mathbf{J} = \begin{bmatrix} \mathbf{J}_1 \\ \mathbf{J}_2 \end{bmatrix}$), then the definition of the coupling matrix \mathbf{X}_{12} corresponds to the notation given in (53):

$$\mathbf{X}_{12} = \mathbf{X}_{\{1,2\}} = \mathbf{J}_{\{1,2\}}^+ - \mathbf{X}_{\{1\}} - \mathbf{X}_{\{2\}} = \mathbf{J}^+ - [\mathbf{J}_1^+ \ \mathbf{J}_2^+] \quad (61)$$

Using this definition, it is now easy to build an inverse $\mathbf{J}^{\oplus \mathbf{H}}$ that fulfills the specification given in Definition 5.1.

Definition 5.3 (Continuous inverse $\mathbf{J}^{\oplus \mathbf{H}}$) Let \mathbf{J} be a matrix of size $(k \times n)$ and \mathbf{H} the corresponding activation matrix whose components $(h_i)_{i \in [1..k]}$ belong to the interval $[0, 1]$. The continuous inverse of \mathbf{J} activated by \mathbf{H} is defined by:

$$\mathbf{J}^{\oplus \mathbf{H}} = \sum_{\mathcal{P} \in \mathfrak{P}(k)} \left(\prod_{i \in \mathcal{P}} h_i \right) \mathbf{X}_{\mathcal{P}} \quad (62)$$

Remark 5.3:

- The continuous inverse of a single-line jacobian is nothing but:

$$\mathbf{J}^{\oplus \mathbf{H}} = h\mathbf{J}^+ \quad (63)$$

where $\mathbf{H} = [h]$.

- The continuous inverse of a double-line jacobian $\mathbf{J} = \begin{bmatrix} \mathbf{J}_1 \\ \mathbf{J}_2 \end{bmatrix}$, activated by \mathbf{H} is:

$$\begin{aligned} \mathbf{J}^{\oplus \mathbf{H}} &= h_1 h_2 \mathbf{X}_{\{1,2\}} + h_1 \mathbf{X}_{\{1\}} + h_2 \mathbf{X}_{\{2\}} \\ &= \begin{bmatrix} h_1 \mathbf{J}_1^+ & h_2 \mathbf{J}_2^+ \end{bmatrix} + h_1 h_2 \mathbf{X}_{12} \end{aligned} \quad (64)$$

This last equation is exactly the preliminary goal written in (59).

Definition 5.3 proposes a new operator to inverse a matrix \mathbf{J} activated by a diagonal activation matrix \mathbf{H} . This operator will now be proved to fulfill the specification given in Definition 5.1.

Theorem 5.1 (Continuity of $\mathbf{J}^{\oplus \mathbf{H}}$) *The inverse $\mathbf{J}^{\oplus \mathbf{H}}$ of \mathbf{J} activated by \mathbf{H} respects the specifications of a continuous inverse given in Definition 5.1.*

Proof: Two points have to be proved. First of all, let us prove that the operator $\mathbf{J}^{\oplus \mathbf{H}}$ is equal to the classical pseudo-inverse $(\mathbf{H}\mathbf{J})^+\mathbf{H}$ when the components of \mathbf{H} are binary (*i.e.* no feature is within the transition region). Let \mathcal{P} be the set of non-zero components of \mathbf{H} . Using the notations of Definition 5.2, we have thus to prove that $\mathbf{J}^{\oplus \mathbf{H}} = \mathbf{J}_{\mathcal{P}}^+$. Using (62), it is possible to write:

$$\mathbf{J}^{\oplus \mathbf{H}} = \mathbf{X}_{\mathcal{P}} + \sum_{\mathcal{Q} \subsetneq \mathcal{P}} \mathbf{X}_{\mathcal{Q}} \quad (65)$$

From (60), it is known that $\mathbf{X}_{\mathcal{P}} = \mathbf{J}_{\mathcal{P}}^+ - \sum_{\mathcal{Q} \subsetneq \mathcal{P}} \mathbf{X}_{\mathcal{Q}}$. Introducing (60) in (65), it is finally possible to obtain $\mathbf{J}^{\oplus \mathbf{H}} = \mathbf{J}_{\mathcal{P}}^+$.

The second point to prove is the continuity of the inverse with respect to the variations of \mathbf{H} . All the coupling matrix \mathbf{X}_p are independent to \mathbf{H} (by Definition 5.2). Thus the inverse (62) is simply a polynomial form of the h_i . Since a polynomial is always continuous, the continuity of the inverse with respect to the variations of \mathbf{H} is demonstrated. \square

5.3 Continuous control law

Based on this new inversion operator, it is possible to propose the following control law:

$$\dot{\mathbf{q}} = -\lambda \mathbf{J}^{\oplus \mathbf{H}} \mathbf{e} \quad (66)$$

Thanks to the nice properties of the continuous inverse, the control law (66) is continuous everywhere. Moreover, when all the features are fully active or fully inactive (*i.e.* $\forall i = 1..k, h_i \in \{0, 1\}$), the control law (66) is equivalent to the classical control law (2) (*i.e.* $\dot{\mathbf{q}} = -\lambda \mathbf{J}_{\mathbf{A}}^+ \mathbf{e}_{\mathbf{A}}$, where $\mathbf{e}_{\mathbf{A}}$ is the active part of \mathbf{e}). It has thus the same stability property of the equivalent task. In particular, if all the features are active at the desired position, the control law (66) is locally stable at task completion.

After having demonstrated in the previous section that none of the classical control ensure the continuity, we have constructed in this section a new control law that ensures the continuity in all cases. In the following section, we will present some experimental results that validate the use of this control law.

6 Experimental results

We present in this section several experiments that study the behavior of the system running the control law presented in the previous sections. The experiments have been realized in simulation, using the classical visual servoing scheme based on feature points [12]. During the execution, some points may leave the camera field of view. They are then removed from the feature set, as done in [11]. In the following, we first recall in Section 6.1 the visual servoing framework. Four typical experiments are then presented in details in the four following subsections.

6.1 Visual servoing implementation

The work presented above is general and could be applied to any robotic task defined by a derivable error \mathbf{e} . In the following, the error function is computed from visual features [12]:

$$\mathbf{e} = \mathbf{s} - \mathbf{s}^* \quad (67)$$

where \mathbf{s} is the current value of the visual features for task \mathbf{e} and \mathbf{s}^* their desired value. In the experiments, the visual features are the 2D positions $\mathbf{p}_i = (x_i, y_i)$ of a set of points in the image. The 3D positions of these points within the camera frame are noted $\mathbf{P}_i = (X_i, Y_i, Z_i)$. The error task corresponds thus to the difference between the current and the desired positions of the points in the image:

$$\mathbf{e} = \begin{bmatrix} x_1 - x_1^* \\ y_1 - y_1^* \\ \dots \\ x_{m/2} - x_{m/2}^* \\ y_{m/2} - y_{m/2}^* \end{bmatrix} \quad (68)$$

where $\mathbf{p}_i^* = (x_i^*, y_i^*)$ is the desired position of the point \mathbf{p}_i in the image, and m is the number of features in \mathbf{e} . The interaction matrix $\mathbf{L}_{\mathbf{s}_i}$ related to \mathbf{s}_i is defined so that $\dot{\mathbf{s}}_i = \mathbf{L}_{\mathbf{s}_i} \mathbf{v}$, where \mathbf{v} is the instantaneous camera velocity. For one point \mathbf{p}_i , $\mathbf{L}_{\mathbf{p}_i}$ is equal to [10]:

$$\mathbf{L}_{\mathbf{p}_i} = \begin{bmatrix} -\frac{1}{Z_i} & 0 & \frac{x_i}{Z_i} & x_i y_i & -(1 + x_i^2) & y_i \\ 0 & -\frac{1}{Z_i} & \frac{y_i}{Z_i} & (1 + y_i^2) & -x_i y_i & -x_i \end{bmatrix} \quad (69)$$

The interaction matrix of the task \mathbf{e} is finally $\mathbf{L} = (\mathbf{L}_{\mathbf{p}_1}, \dots, \mathbf{L}_{\mathbf{p}_{m/2}})$. From (67), it is clear that the interaction matrix \mathbf{L} and the task Jacobian \mathbf{J} are linked by the relation:

$$\mathbf{J} = \mathbf{L} \mathbf{M} \mathbf{J}_q \quad (70)$$

where the matrix \mathbf{J}_q denotes the robot Jacobian ($\dot{\mathbf{r}} = \mathbf{J}_q \dot{\mathbf{q}}$) and \mathbf{M} is the matrix that relates the variation of the camera velocity \mathbf{v} to the variation of the chosen camera pose parametrization ($\mathbf{v} = \mathbf{M} \dot{\mathbf{r}}$).

It is possible to use an approximation $\hat{\mathbf{J}}$ of \mathbf{J} in order to improve the behavior of the control law. In particular, different choices are available for \mathbf{L} [14]. We choose $\hat{\mathbf{J}} = \mathbf{L}^* \mathbf{M} \mathbf{J}_{\mathbf{q}}$, where \mathbf{L}^* is the interaction matrix computed at the desired position. This choice is frequently done since it provides better camera trajectories and reduces the risk of falling in local minimum. However, due to this approximation, some points may leave the camera field of view during the servo [4]. These points are then removed from the feature set as proposed in [11]. The error task is a varying-feature-set task based on (67):

$$\mathbf{e}_{\mathbf{q}} = \mathbf{H} \mathbf{e} \quad (71)$$

where \mathbf{H} is a diagonal matrix. The coefficients h_i of \mathbf{H} are defined by:

$$h_{2i} = h_{2i+1} = \min(h_x(x_i), h_y(y_i)) \quad (72)$$

The horizontal activation function h_x is defined by:

$$h_x(x) = \begin{cases} 1 & \text{if } \bar{x}^- + \epsilon_x \leq x \leq \bar{x}^+ - \epsilon_x \\ 0 & \text{if } x \geq \bar{x}^+ \text{ or } x \leq \bar{x}^- \\ f_{\epsilon_x} \left(x - (\bar{x}^+ - \epsilon_x) \right) & \text{if } \bar{x}^+ - \epsilon_x \leq x \leq \bar{x}^+ \\ f_{\epsilon_x} \left((\bar{x}^- + \epsilon_x) - x \right) & \text{if } \bar{x}^- \leq x \leq \bar{x}^- + \epsilon_x \end{cases} \quad (73)$$

where $[\bar{x}^-, \bar{x}^+]$ is the horizontal range of the image, ϵ_x tunes the length of the transient interval, and the transient function f_{ϵ} is defined by:

$$f_{\epsilon}(x) = \frac{1}{2} \left(1 + \tanh \left(\frac{\epsilon}{x} - \frac{1}{1 - x/\epsilon} \right) \right) \quad (74)$$

so that $f_{\epsilon}(0) = 0$ and $f_{\epsilon}(1) = 1$.

The vertical activation function h_y is defined similarly. The activation function is plotted in Fig. 3. We have chosen to use this function rather than the one originally proposed in [11] in order to have a full activation $h = 1$ in the control part of the image (which is not the case in [11]). For the experimental setup, this point is very important in order to test the behavior of the control scheme at full activation².

6.2 First experiment: non-redundant task

The first experiment has been realized using a two-point target. The dimension and the rank of the task are thus both equal to 4 at full activation: the task is always non-redundant. At the initial position, one point is out of the field of view. We mainly consider the continuity of the control law when the point enters the field of view. The experiment is summed up in Figures 4, 5, 6, 7 and 8.

²Nevertheless, if needed or wanted, an activation function similar to the one proposed in [11] can be obtained by simply setting ϵ_x and ϵ_y to 1.

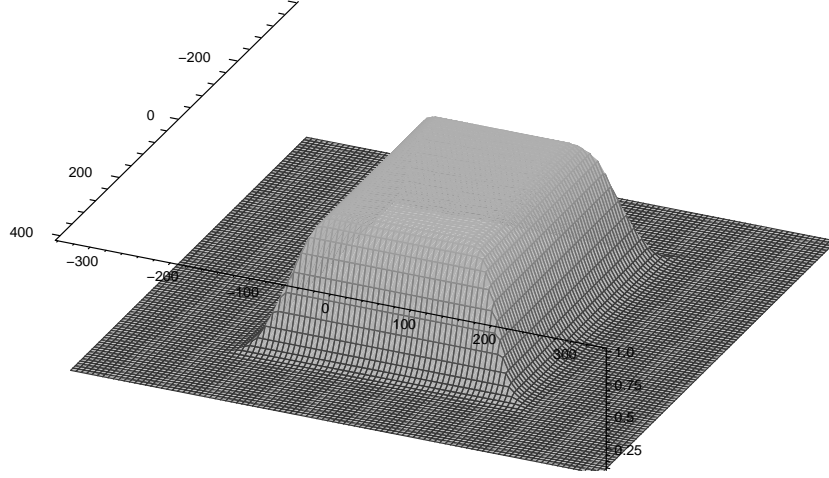


Figure 3: Activation function (72) defined with respect to image point coordinates. This function is null when the corresponding point leaves the field of view. It is equal to 1 when the point is far enough from the image borders. The function has been plotted for $\bar{x}^+ = -\bar{x}^- = 320(\text{pixels})$, $\bar{y}^+ = -\bar{y}^- = 240$ and $\epsilon_x = \epsilon_y = 50$.

As proved in Section 4.3, the two control laws using respectively a partial approximation (27) and no approximation (28) of the activation matrix provide similar results (see Fig. 4), except for small numerical approximations. In both cases, a large discontinuity occurs when the point is activated. The acceleration of the control law (28) is given in Fig. 5. The acceleration peak is clearly visible.

The continuity of control laws (27), (28), (29) and (66) are compared in Fig. 6. Only the two last ones are able to ensure the continuity. Figures 6(c) to (e) point out the importance of the threshold η of the damped-least-square operator. When η is too small (Fig. 6(c)), the same peak of acceleration as with (28) is obtained. The more η increases, the more the acceleration peak decreases. When η is large enough, the peak completely disappears (Fig. 6(d)). However, the control law behavior is then very close to the one obtained with the transpose operator, and the convergence becomes very slow. In particular, in the experiment we are considering here, the servo is even not able to converge (see Fig. 8). A good compromise for this case is obtained for $\eta = 1e^{-3}$. However, it can be shown experimentally that the satisfactory value really depends on the feature values and on the jacobian. In particular, it seems to be very correlated to the singular values of the Jacobian. It is thus very difficult to find a correct value for any experimental conditions.

Finally, the new control law we propose, (66), provides a very good behavior. The control law is continuous (see Fig. 7). The acceleration is similar to the one obtained with (29) for the proper tuning of ϵ . The improvement using (66) is also visible on Fig. 8, where the trajectories of the two image points during the execution are plotted. With the control law

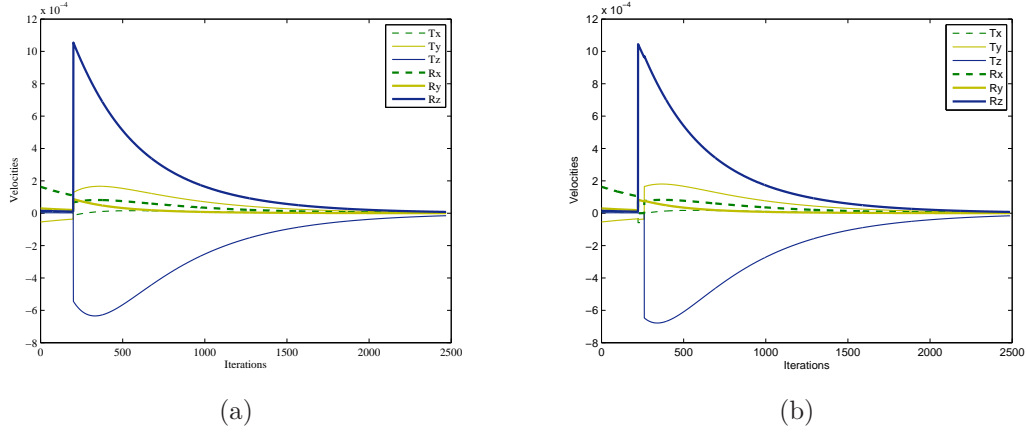


Figure 4: Experiment 1: comparison of the velocities sent to the robot when using the control law with a partial approximation (27) on Fig. (a), and without approximation (28) on Fig. (b). The two behaviors are very similar. The discontinuity at iteration 220 corresponds to the entry of the second point within the camera field of view. On the second figure, two discontinuities appear at iteration 220 and 250, and not only one, like on (a). This is due to numerical approximations in the computation of the pseudo inverse.

(27) (trajectory in blue), an inflection point in the trajectory corresponding to a discontinuity of the derivative is observed as soon as the point enters the camera field of view. The non-convergence of the control law (29) (no approximation on \mathbf{H}) is clearly observable in red. With our new control law (66), the point trajectory is smooth. Its trajectory is changed within the transient interval, with respect to the continuous variation of \mathbf{H} .

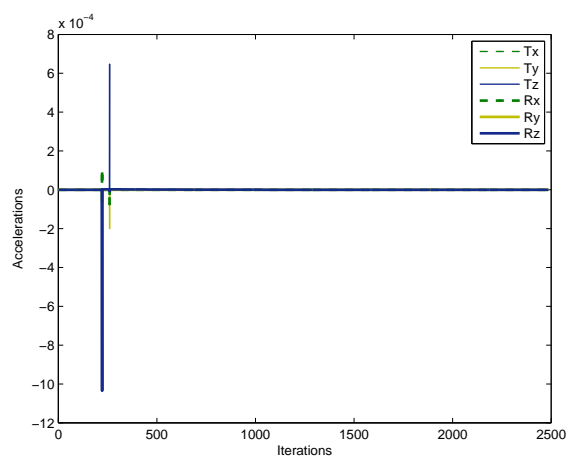


Figure 5: Experiment 1: Robot accelerations using control law (28). Strong peaks of acceleration appear when a feature is activated (iterations 220 and 250).

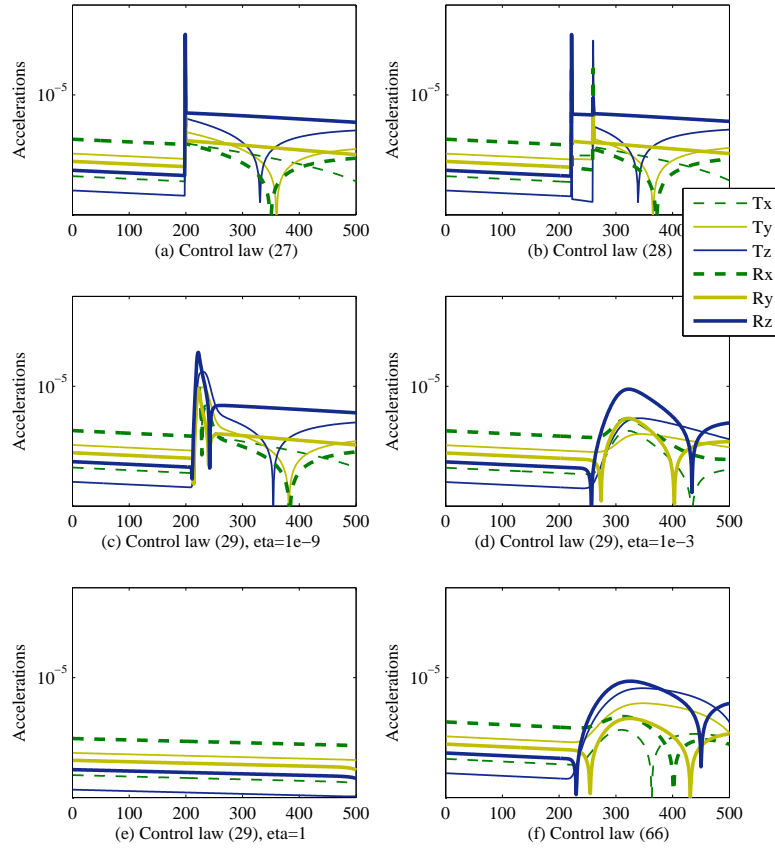


Figure 6: Experiment 1: Comparison of the acceleration peaks when using control laws (27) on (a), (28) on (b), (29) with several tuning on graphs (c) to (e) ($\eta = 1e^{-9}$, $1e^{-3}$ and 1) and (66) on (f). Since the main relevant situation occurs when the second point enters the camera field of view at iteration 200, the plots have been limited to the interval $[0, 500]$. Control laws (27) and (28) are unable to ensure the continuity. Concerning (29), the greater is the tuning parameter η , the smaller is the acceleration peaks. A good compromise is obtained for this experiment with $\eta = 1e^{-3}$ (with $\eta = 1$, the behavior seems to be correct while looking the continuity, but Fig. 8 shows that it is not when considering the overall task). Control law (66) also provides a smooth control, with no peak of acceleration.

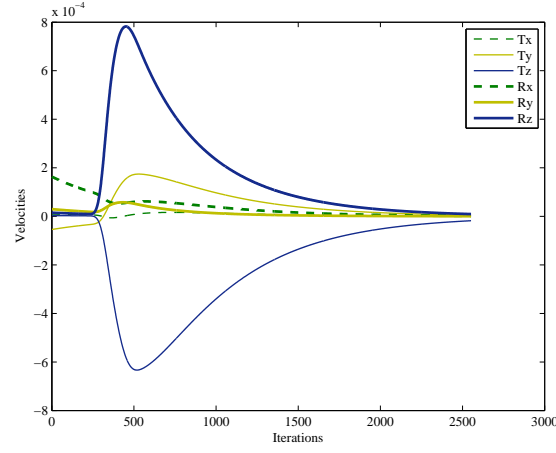


Figure 7: Experiment 1: velocities sent to the robot using the control law with the continuous inverse operator (66). The control law is continuous.

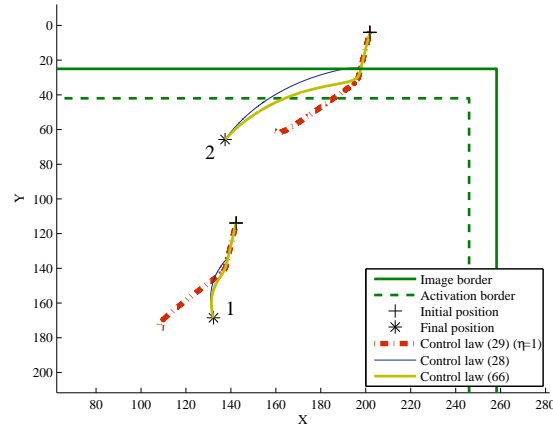


Figure 8: Experiment 1: points trajectories in the image. At the beginning of the servo, Point 2 is out of the image. The two desired positions are within the full-activation area. Control law (29) using a too large value of η (in red), even if the velocity is continuous, is unable to converge and thus to achieve the task. On the opposite, control laws (28) (in blue) and (66) (in green) manage to reach the desired position. Control law (28) provides an image trajectory which is quite abrupt when the point enters the activation area. On the opposite, the trajectories using control law (66) are smooth.

6.3 Second experiment: oscillations with a non-redundant task

This experiment illustrates the problems that may occur if the control law is discontinuous. In the case presented below, the control law oscillates when a point leaves the field of view, which could even avoid the convergence. This problem is solved when using a continuous control law. The experiment is summed up in Figures 9, 10, 11, 12 and 13.

The target used in this experiment is the same as for Experiment 1. Two points are considered, which gives a task of dimension and rank 4. The displacement required now is mainly a large rotation around the optical axis. During the execution of this rotation, one point (\mathbf{P}_2) gets close to the activation border (see Fig. 9). At this point, if the two points are considered within the minimization scheme, the computed control, mainly a \bar{z} rotation, makes \mathbf{P}_2 leave the camera field of view. On the opposite, if we consider only the point that remains within the image, \mathbf{P}_1 , the computed control becomes mainly a pan-tilt motion that tends \mathbf{P}_2 to enter again the activation zone. The oscillation observed is thus due to this dilemma: if \mathbf{P}_2 is inactivated, it enters the image which activates it, which makes it leaves the image, etc.

The oscillation occurs as soon as the control law is not continuous. Control laws (27), (28) and (29) (if the tuning parameter ϵ is too small) oscillate, as can be seen on Fig. 10. Moreover, control law (28) is unable to leave the oscillation area and the servo does not converge.

These oscillations within the control law are due to oscillations of the coefficients of the activation matrix. The evolution of the activation coefficient corresponding to \mathbf{P}_2 is plotted in Fig. 11. When using control law (28), the coefficient oscillates and finally converges to a value different from 1: \mathbf{P}_2 is then fixed near the image border, and the servo does not converge. With the control law (29), the activation coefficient oscillates around a value that depends on the tuning parameter η . When η is large enough, the oscillation disappears, and the control law is continuous. Experimentally, it has been noticed that the oscillations disappears when $\eta > 1e^{-3}$.

A comparison of the accelerations using a proper tuning in (29) or using (66) is presented in Fig. 12. As before, the larger is η , the smoother is the control. The control law (66) provides a control as smooth as any control obtained with different tunings of (29). The velocities sent to the robot are smooth, as it can be seen in Fig. 13.

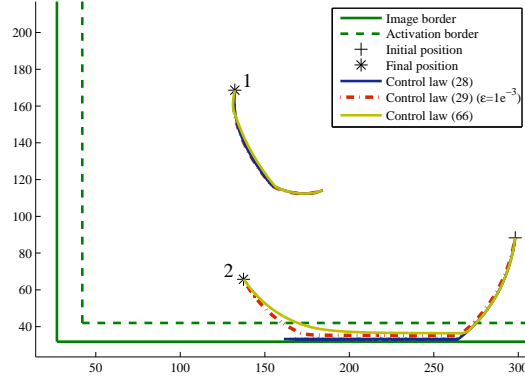


Figure 9: Experiment 2: image point trajectories. The main required motion is a rotation around the optical axis. With the classical control law (28), Point \mathbf{P}_2 leaves the image. The discontinuities of the control law at rank loss produce then oscillations that are visible in Point \mathbf{P}_2 trajectory. Point 2 oscillates around the image border and the servo is even unable to converge. With an appropriate tuning of control law (29) or with the continuous control law (66), the servo converges.

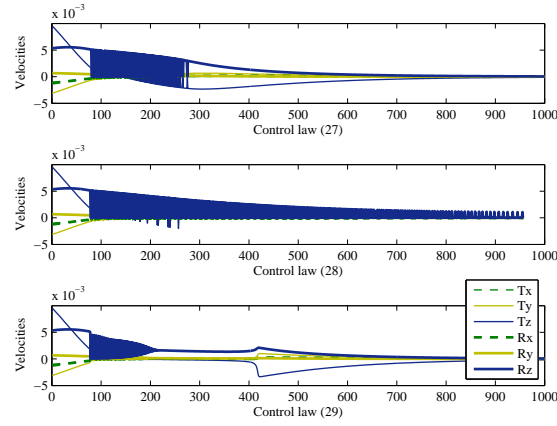


Figure 10: Experiment 2: velocities sent to the robot using control laws (27) without approximation, (28) with a partial approximation and (29) with the damped least square inverse operator. Oscillations appear when \mathbf{P}_2 reaches the image border (at iteration 100).

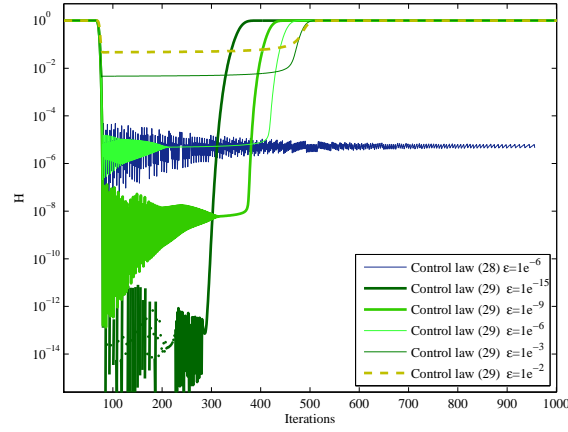


Figure 11: Experiment 2: activation coefficient corresponding to \mathbf{P}_2 , using control law (28) and several different η for control law (29). With (28), the activation coefficient oscillates and stabilizes to a value different from 1. With (29), the coefficient oscillates around a value that depends on η , but finally succeeds to converge toward 1, which makes the point enter the camera field of view. When η is large enough, the oscillations disappear.

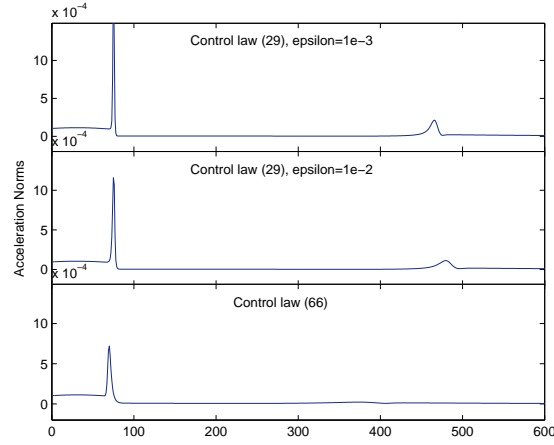


Figure 12: Experiment 2: comparison of the acceleration of the control laws (29) and (66). The larger η is, the smoother the control computed from (29) becomes. The control law (66) provides a smoother acceleration, and does not require the tuning of a particular parameter.

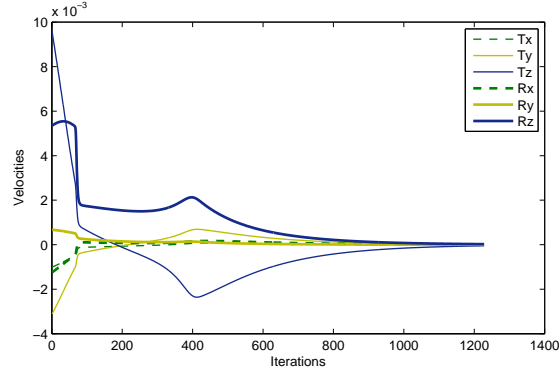


Figure 13: Experiment 2: velocities sent to the robot using control law (66). The control law is continuous.

6.4 Third experiment: full-redundant task

In this experiment, the task stays full redundant during all the execution. The task is defined by seven image points. The image point trajectories are given on Fig. 14-(a) along with the camera trajectory on Fig. 14-(b). These graphs show that, since the task is always full redundant, all the control laws provide similar behaviors, except the first one, with partial approximation (27), as expected. As long as there is no discontinuity to smooth, the use of the damped least-square inverse in (29) does not bring significant modification here.

On Fig. 15, comparing the velocities sent to the robot for the two last control laws, it appears that our new inverse operator is equivalent to the classical one when no discontinuity appears. The condition of equivalence required within the formal definition of the desired continuous inverse operator in 5.1, seems to be (experimentally speaking) respected.

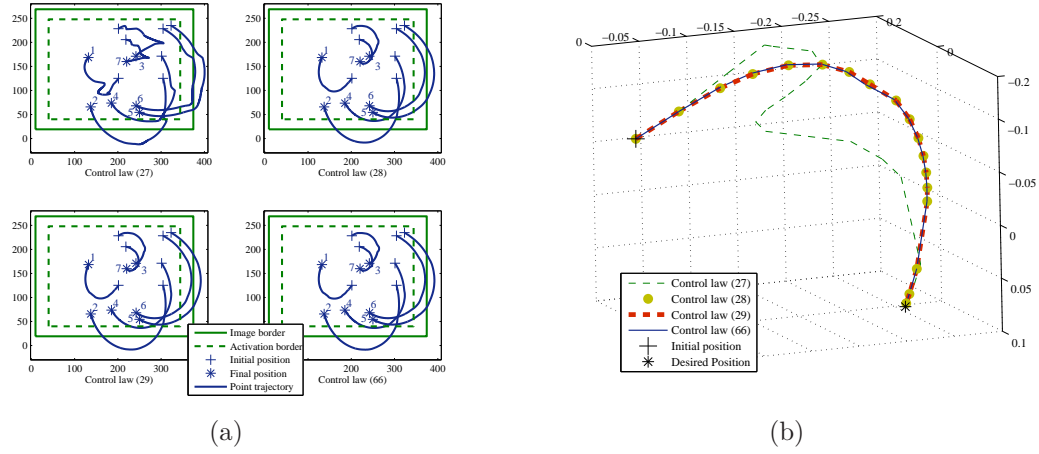


Figure 14: Experiment 3: (a) point trajectories in the image for control laws (27) (partial approximation), (28) (no approximation), (29) (no approximation and damped-least-square inverse operator) and (66) (continuous inverse operator), (b) associated 3D camera trajectories. The behavior appears to be abrupt with (27), in comparison with the trajectories obtained with any of the three other control laws that are shown to be smoother.

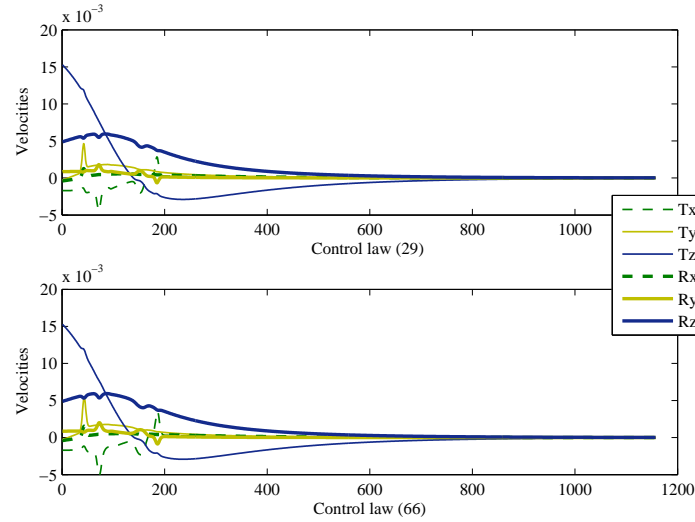


Figure 15: Experiment 3: comparison of the velocities computed from (29) and (66). The velocities are almost equal: the two control laws can be considered equivalent.

6.5 Fourth experiment: redundant task

This last experiment gathers the different situations considered separately in the two previous ones. Eight points are considered. The complete task is thus full redundant. The desired motion is mainly a rotation around the optical axis \vec{z} . While performing this movement, a large amount of points initially close to the image borders leaves the camera field of view, making the task being just redundant and then not redundant (between iterations 150 and 500, only two points remain within the camera field of view). Finally, at the end of the execution, all the points re-enter the image frame.

Let us first consider the behavior of the control law without any approximation of the activation matrix. The image point coordinates are displayed on Fig. 16, while the associated camera velocities are displayed on Fig. 17. Only two points are active from iteration 150, and the task is then non-redundant. When \mathbf{P}_3 enters the camera field of view (around iteration 500), its activation involves oscillations within the velocity. Despite this behavior, \mathbf{P}_4 succeeds in entering the camera field of view. The task is then fully redundant, and the smoothing behavior is taken into account, which stops the oscillations. When observing these two graphs, several discontinuities can be observed on the image point trajectories, always corresponding to abrupt changes of camera velocities.

Using the damped least-square inverse operator, in control law (29), and with a good tuning of η , it is possible to obtain an overall better behavior. Fig. 18 shows that it is possible to reduce significantly the acceleration observed around iteration 150.

Nevertheless, as already stated, it is not easy to find the good value of η , and that is why our proposed operator is appealing since it does not depend on any parameter tuning. On Fig. 18, one can see that the accelerations observed are even smaller than the one obtained with the best tuning of η found. Fig. 19 shows that the point trajectories are much smoother than with control law (29). One can then easily imagine that the camera velocities are much more continuous than the one observed on Fig. 17. This is effectively verified on Fig. 20, where the camera velocities of control laws (29) and (66) are compared. The oscillations of the previous experiment are no more present even if the two graphs are quite similar. Indeed, as shown in Section 5, the two control laws are equivalent when no point is being activated or inactivated. Nevertheless, the new inverse operator provides a smoother behavior when the system is not full-redundant, as it can be seen at iteration 200 for example.

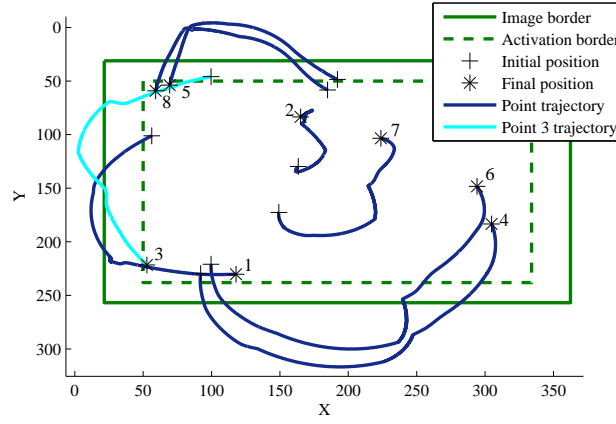


Figure 16: Experiment 4: point trajectories using control law (28). When \mathbf{P}_3 enters the image, the control law oscillates. \mathbf{P}_3 then oscillates around the activation border

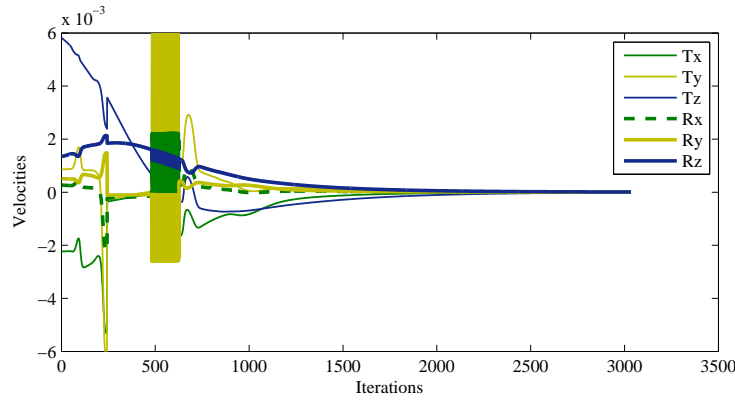


Figure 17: Experiment 4: velocities sent to the robot using control law (28). The control law is continuous until the fourth point leaves the field of view (iteration 230). The task is then non redundant, and the continuity that \mathbf{H} is supposed to provide is not ensured any more. The control law is then discontinuous. When \mathbf{P}_3 enters the image (iteration 500), the control law oscillates. These oscillations stop when another point enters the image (around iteration 650). The task is then full-redundant, and the control law is continuous again.

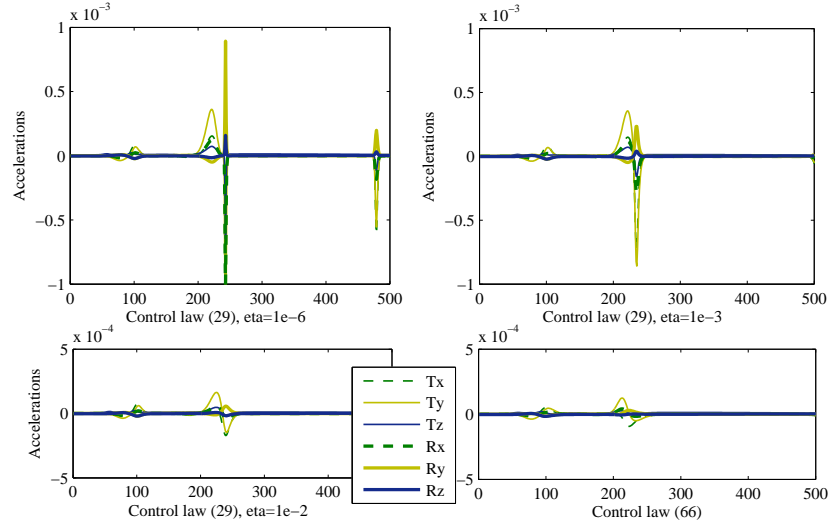


Figure 18: Experiment 4: comparison of the accelerations observed when using control law (29) with different values of η and using control law (66).

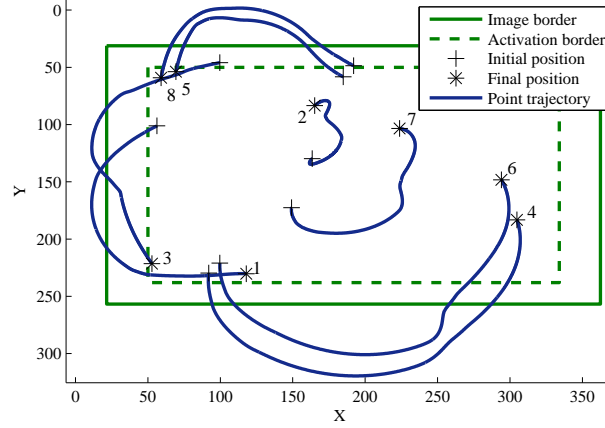
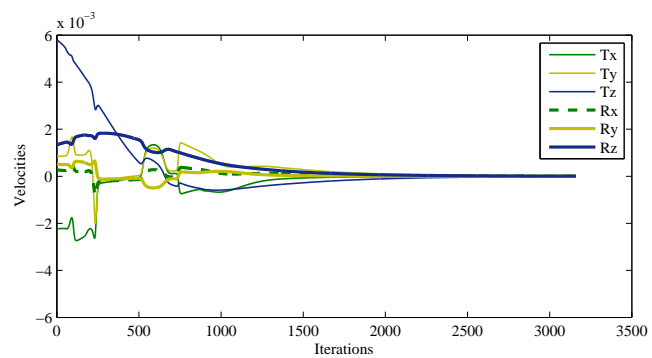
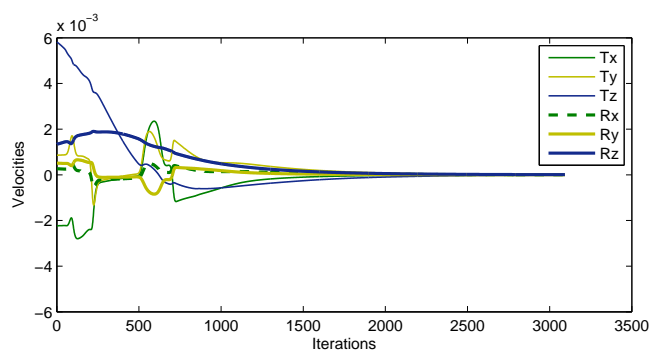


Figure 19: Experiment 4: point trajectories using control law (66). The trajectories are smooth. The general form of the trajectories is similar to the one obtained in Fig. 16, except when a point is close to the image border. In these situations, the point trajectories are smoother. It has also to be noticed that \mathbf{P}_3 does not oscillate when activated again.

(a) Control law (29), $\eta = 1e^{-2}$ 

(b) Control law (66)

Figure 20: Experiment 4: Comparison of the control laws (29) and (66). The second control law appears much smoother during the time interval $[200, 650]$, where the task is non redundant.

7 Conclusion

This study has considered robotic tasks defined by a set of features whose dimension is varying along the time. Such tasks have been addressed in literature for considering several applications. Classically, the variation of the size of the feature set is performed by using an activation matrix whose value varies between 0 and 1. Within this context, a natural interrogation concerns the behavior of the resulting control law at the critical point of activation or inactivation of a feature. We have studied this situation, through several combinations and approximations of the jacobian associated to such varying feature set.

The main point of this theoretical study is that the use of an activation matrix with the classical pseudo inverse is inefficient to ensure the continuity when the input signal is not fully redundant. More precisely, it has been proved that the pseudo inverse is always discontinuous at the activation of a non redundant feature except in the particular case of a perfect decoupling.

To deal with this problem, an original inverse operator has been introduced, that insures the continuity even when the rank of the jacobian changes. This new inversion operator has then been used to define a new control law, continuous everywhere. The good properties of this control law have then been verified on several experiments.

A Classical control laws versus varying feature sets

This section investigates the behavior of the activation-based control laws presented in Section 2 when the number of visual features used varies. More precisely, the critical situation corresponding to a change of rank due to the activation or the inactivation of a feature is deeply studied. Five different simulations, considering a six DOF robot, are presented. The qualitative servo scheme (recalled in Section 2.5) is considered in all experiments: a classical servo scheme is used when the features are outside the confident interval $[-1, 1]$. A feature is inactivated as soon as it enters the confident interval. The transient parameter ϵ that tunes the smoothness of the activation matrix is set to 0.1: the two transient zones are thus $[1, 1.1]$ and $[-1.1, -1]$.

In the first experiment, control laws (28) and (29) are compared in order to emphasize the interest of the damped-least-square operator. The behavior of the three control laws (27), (29) and (30) are then compared when considering a non-redundant task, a full redundant task and a redundant one. Finally, the case of a perfect decoupled feature is studied to prove experimentally the result obtained in Theorem 4.3.

Note that the feature values and the jacobian are initially randomly set. This means that some features can be incompatible, making the null error unattainable. Nevertheless, these experiments remain interesting since they shed the light on the control law behaviors.

A.1 First experiment: comparing $(\mathbf{HJ})^+$ and $(\mathbf{HJ})^\dagger$

In this first experiment, the task chosen is such that some features are activated or inactivated during the simulations. Two executions are compared. The first one uses the classical pseudo-inverse operator \mathbf{A}^+ (control law (28)) and, the second one uses the damped-least-square-inverse operator \mathbf{A}^\dagger (control law (29)). The corresponding experiments are described by Figures 21 and 22. In each of them, the upper figure presents the variations of the feature values, while the lower one presents the velocity sent to the controller.

Figure 21 clearly presents the discontinuities that occur with the classical inverse operator. Peaks observed correspond to the activation or inactivation of a feature. It can be also observed that the system is very oscillatory before and after each activation or inactivation. These oscillations are due to the instability of the matrix inversion when its rank varies. In the pseudo-inverse, a very thin threshold is used for considering if a singular value is null or not (as usually, the threshold is set to $1e^{-6}$). A singular value remains close to the frontier during a non negligible time (being higher and lower this threshold), which induces these oscillations and peaks.

It is clear on Fig. 22 that (29) is much more stable. The oscillations observed previously, and most of the previous peaks are not present anymore. Nevertheless, some large accelerations at feature activation or inactivation remain present.

Since (29) appears to be much more stable in practice, and since it has been proved to be equivalent to (28) with respect to continuity properties, only (29) will be used in the following experiments.

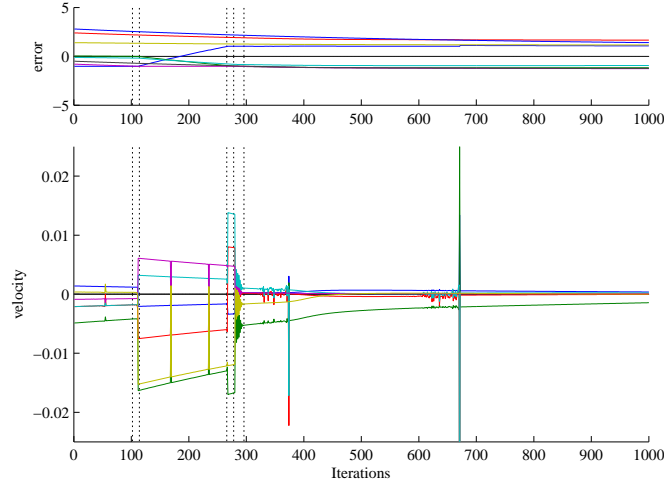


Figure 21: Exp. 1: comparison of $(\mathbf{HJ})^+$ and $(\mathbf{HJ})^\dagger$. Control law (28) which uses the classical pseudo-inverse operator \mathbf{A}^+ is considered. up : error on features coordinates, down: velocity computed. We recall that the feature confident interval (*i.e.* inactivation interval) is $[-1, 1]$, and the transient threshold is set to 0.1. Vertical dotted lines denote a point activation or inactivation. The velocity is discontinuous and oscillatory.

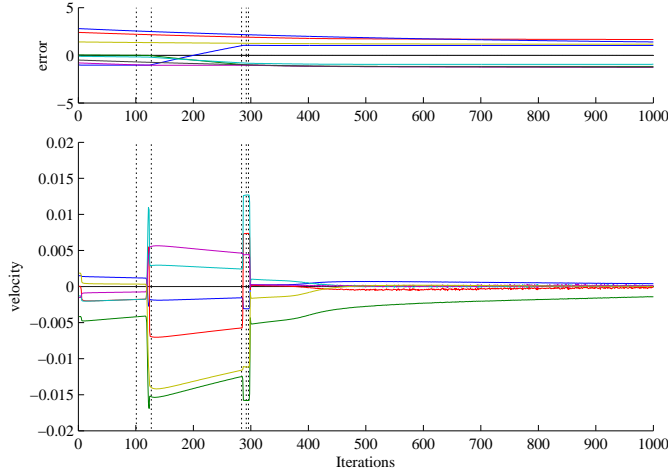


Figure 22: Exp. 1: comparison of $(\mathbf{HJ})^+$ and $(\mathbf{HJ})^\dagger$. The control law (29), that uses a damped-least-square-inverse operator \mathbf{A}^\dagger is illustrated here. Oscillations previously observed are not present anymore, but discontinuities remain at feature activation/inactivation.

A.2 Second experiment: non-redundant task ($\dim \mathbf{e} = 6$)

The task considered here is composed of six features that are used to control the six DOF of the robot. The jacobian (\mathbf{HJ}) is always FRR.

This experiment compares the execution for the same task obtained with control laws (27), (29) and (30). Respective results are presented on Fig. 23, 24 and 25. Each execution is described by four graphs, respectively corresponding to the evolution of the input features e_i during the execution, the control law results, the associated acceleration, and finally the number of active features used to update the control law. The confident interval remains identical to the one used previously (*i.e.* $[-1, 1]$, and $\epsilon = 0.1$).

At the beginning of the motion, only three of the six features are active, since some of them already belong to the confident interval. The inactive features reach out of the confidence interval during the minimization of the active part of the error. The activations occur at iterations 100, 200 and 220.

One can easily observe that the three executions are almost identical. The activation of the initially inactive features occur at the same iteration. The discontinuities of the three control laws are obvious. Each activation produces a strong acceleration.

This experiment validates the theorem result presented in (38): when the task is FRR, the smoothing effect of \mathbf{H} is indeed not taken into account within the control law, whatever control law considered. Even if the use of the damped-least-square inverse operator in (29) induces a smoothing effect, this one seems to be in this situation negligible, and is clearly insufficient to obtain a continuous behavior.

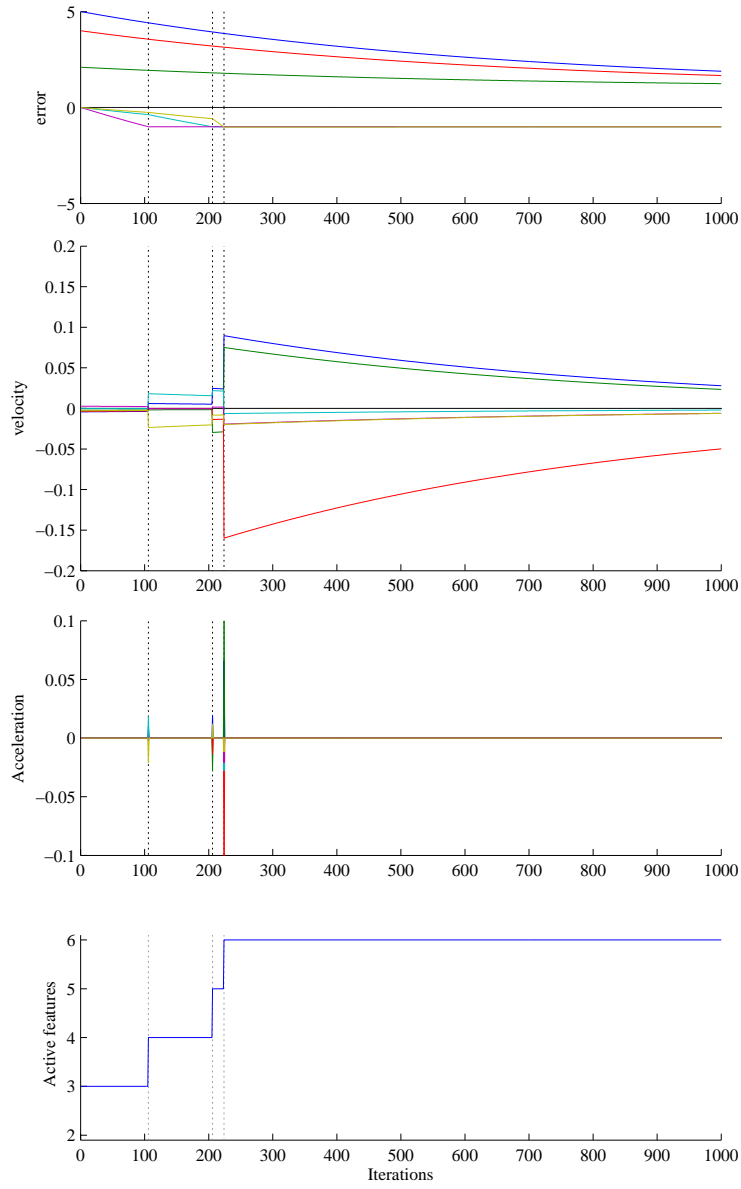


Figure 23: Exp. 2 ($\dim \mathbf{e} = 6$): control law (27), $\dot{\mathbf{q}} = -\lambda(\hat{\mathbf{H}}\mathbf{J})^+ \hat{\mathbf{H}}\mathbf{e}$, is used. A peak of acceleration appears each time a feature is activated or inactivated.

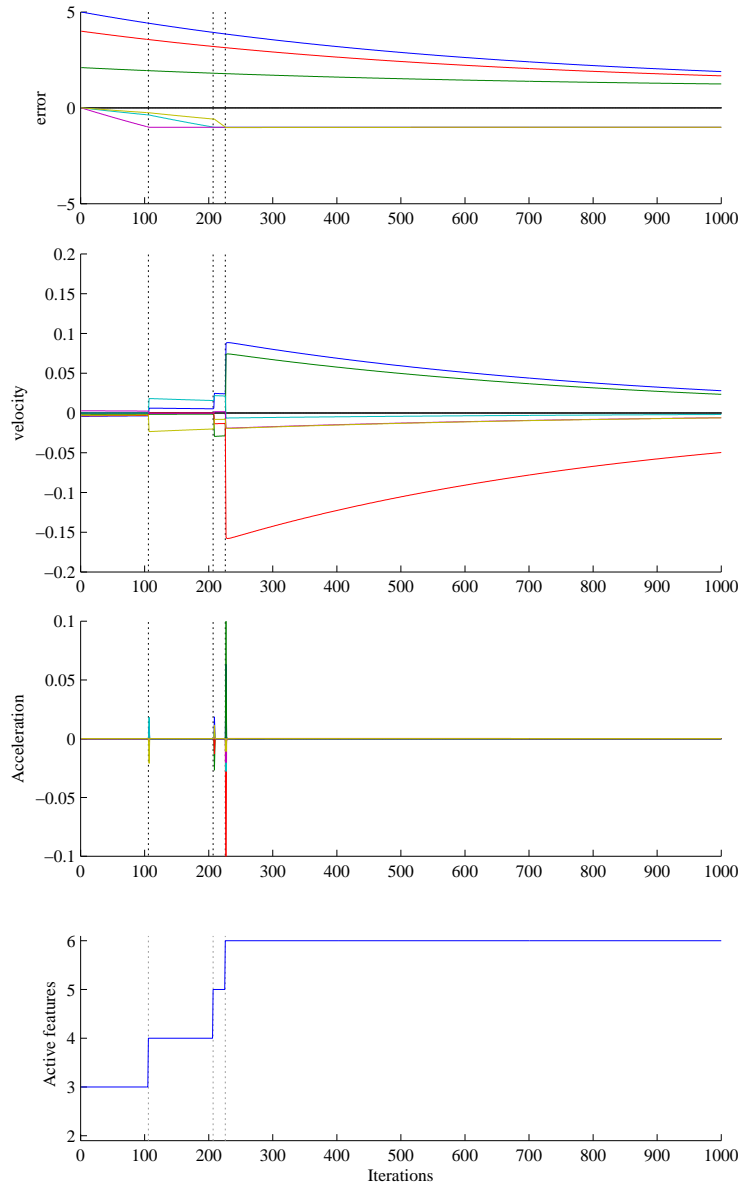


Figure 24: Exp. 2 ($\dim \mathbf{e} = 6$): control law (29), $\dot{\mathbf{q}} = -\lambda(\mathbf{H}\mathbf{J})^\dagger \mathbf{H}\mathbf{e}$, is used. A peak of acceleration appears each time a feature is activated or inactivated. This execution is very similar to the one previously presented, obtained using (27). Particularly, the continuity of the velocity is not better than when using the approximated control law (27).

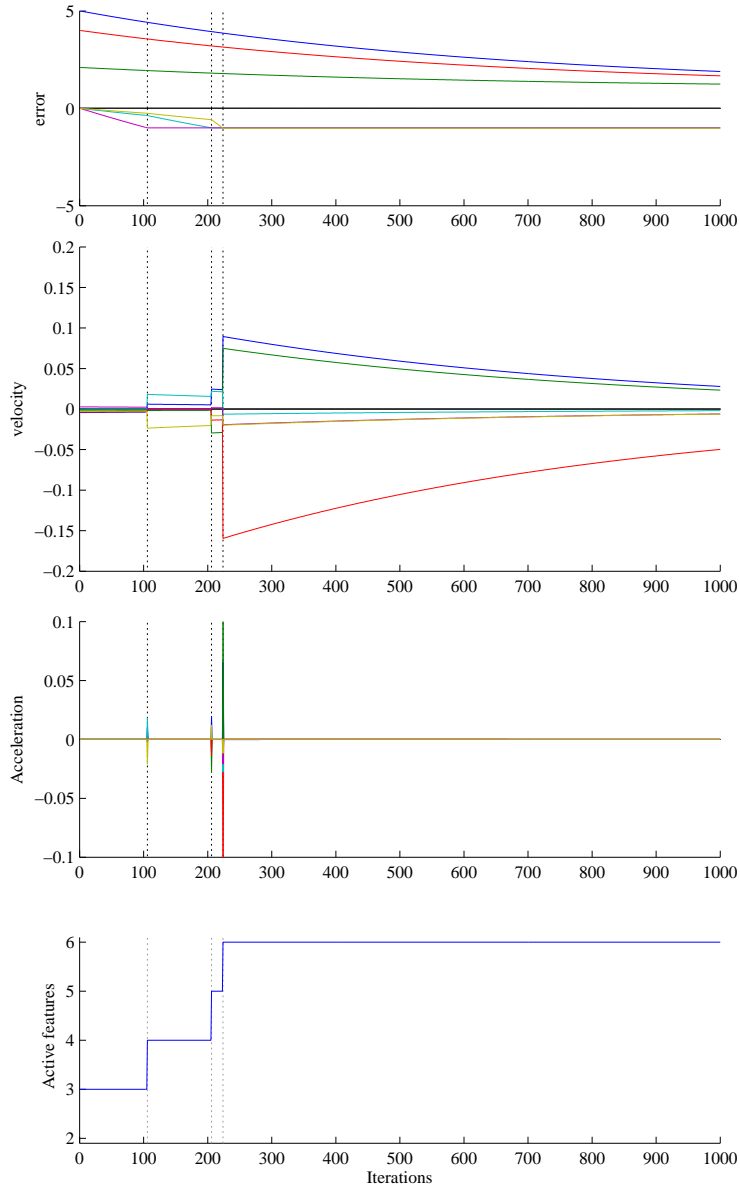


Figure 25: Exp. 2 ($\dim \mathbf{e} = 6$): control law (30), $\dot{\mathbf{q}} = -\lambda(\mathbf{H}\mathbf{J})^+ \widehat{\mathbf{H}}\mathbf{e}$ is used. The execution is similar to the two previous ones. The same discontinuities are observed.

A.3 Third experiment: full-redundant task ($\dim \mathbf{e} = 8$)

In this experiment, two features have been added to the task. These two features are coupled to each of the other ones: the task has thus a full-redundant input. During the task execution, the number of activated features is always larger than or equal to 6 and the task has always a full-redundant input.

Thus, when adding or removing a feature, the hypothesis of (38) is theoretically never respected. These experiments confirm this fact.

Like in the previous experiment, the three control laws are compared, and resulting graphs are presented on Figures 26, 27 and 28.

The main conclusion of these experiments is that the use of an activation matrix is here effective. The control law (29), studied on Fig. 27 is shown to be continuous, no acceleration peak occurs. This is not the case for control laws that use no progressive activation (control law (27)) or only partial activation (control law (30)). As it can be seen on Figures 26 and 28, discontinuities in the velocity and acceleration peaks occur at each activation or inactivation of a feature.

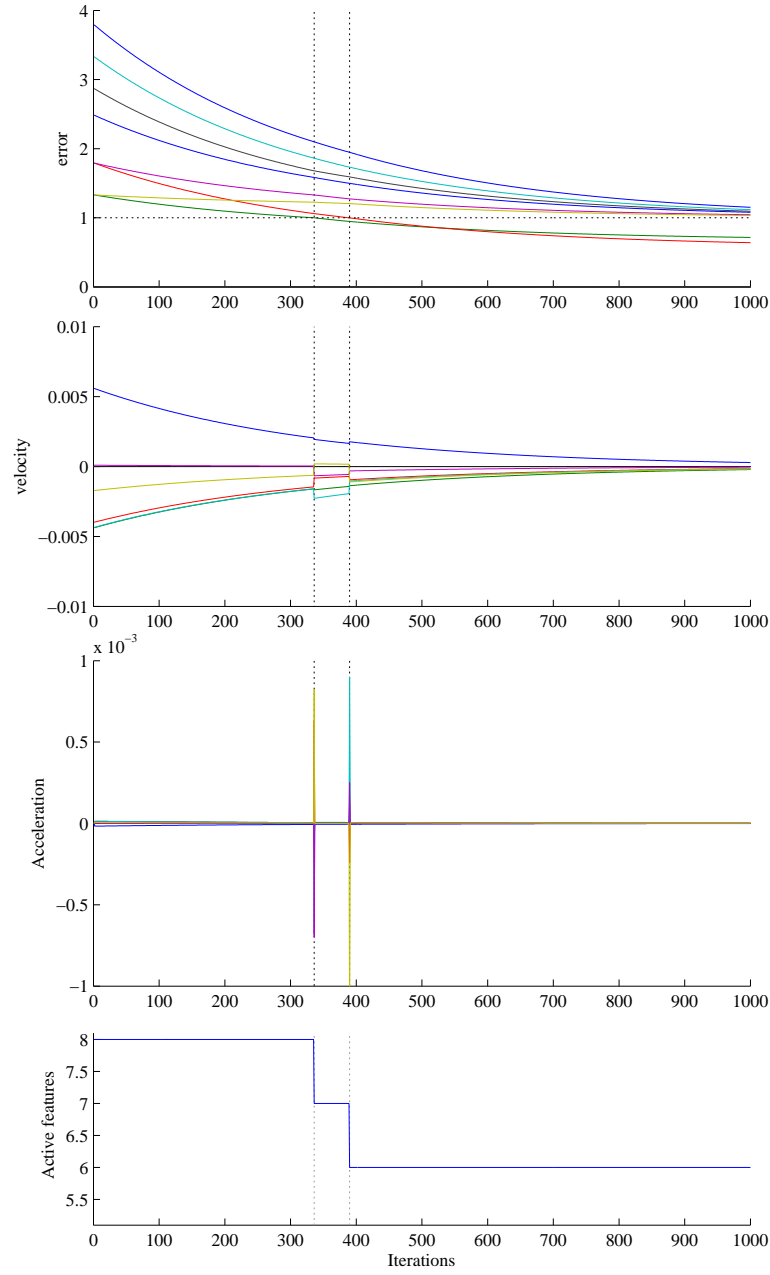


Figure 26: Exp. 3 ($\dim \mathbf{e} = 8$, the input features are always full redundant): control law (27), $\dot{\mathbf{q}} = -\lambda(\hat{\mathbf{H}}\mathbf{J})^+ \hat{\mathbf{H}}\mathbf{e}$, is used. The control law is not continuous, and a peak of acceleration appears each time a feature is activated or inactivated.

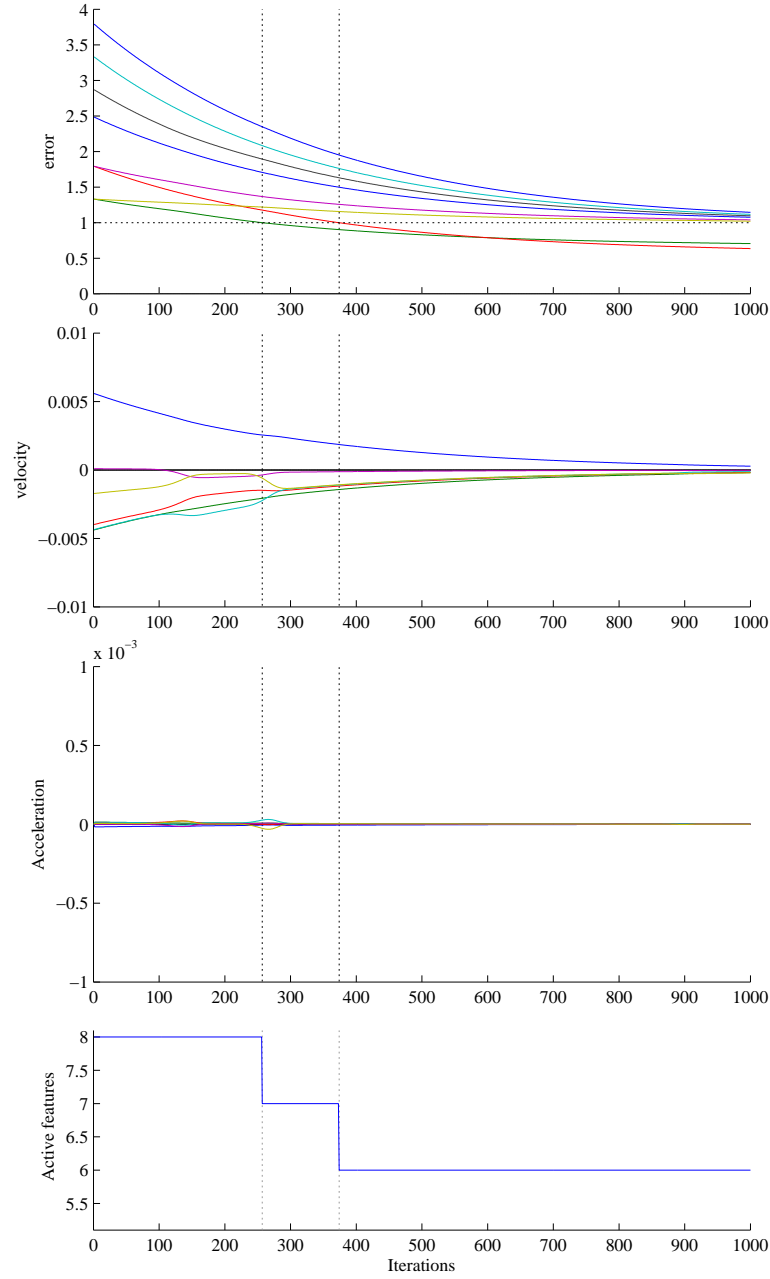


Figure 27: Exp. 3 ($\dim \mathbf{e} = 8$, the input features are always full redundant): control law (29), $\dot{\mathbf{q}} = -\lambda(\mathbf{H}\mathbf{J})^\dagger \mathbf{H}\mathbf{e}$, is used. The control law is continuous, and no peak of acceleration appears during the servo.

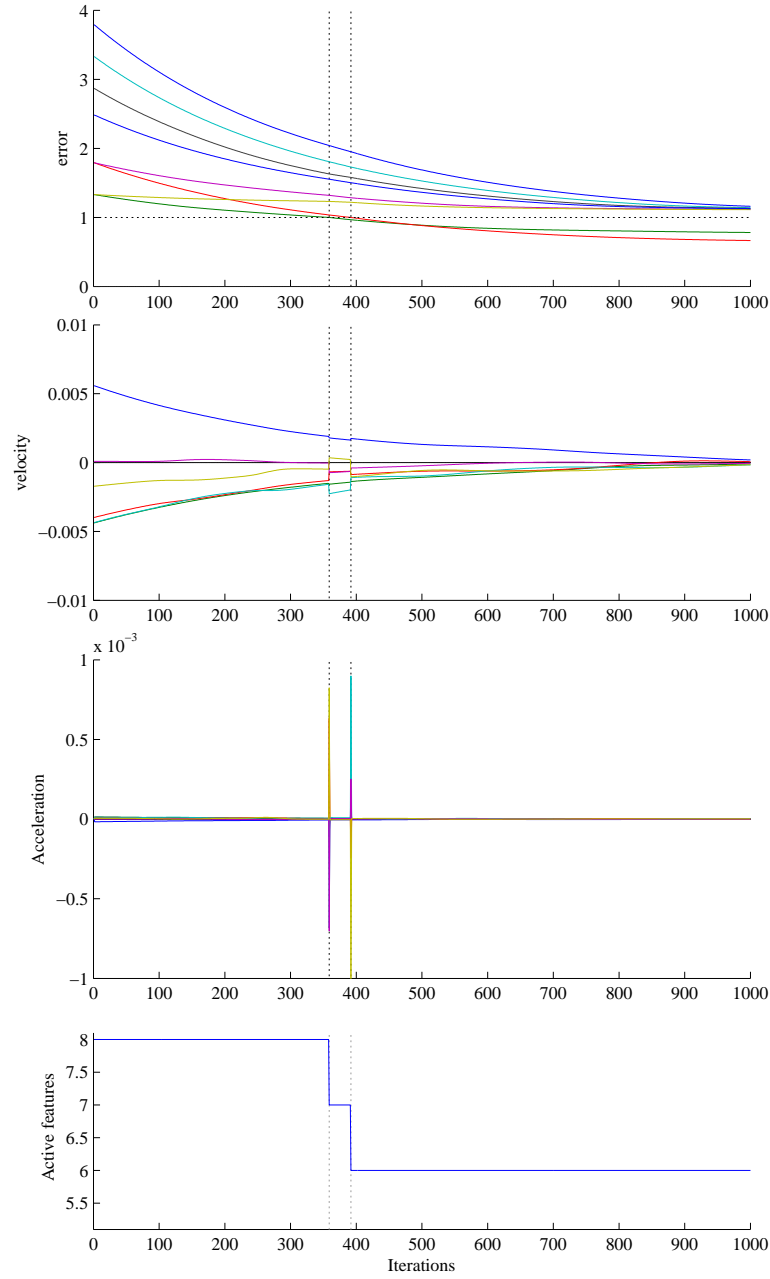


Figure 28: Exp. 3 ($\dim \mathbf{e} = 8$, the input features are always full redundant): control law (30), $\dot{\mathbf{q}} = -\lambda(\mathbf{H}\mathbf{J})^+ \hat{\mathbf{H}}\mathbf{e}$, is used. The control law is not continuous, and a peak of acceleration appears each time a feature is activated or inactivated.

A.4 Fourth experiment: redundant (not full redundant) input ($\dim \mathbf{e} = 12$)

In this experiment twelve input features are considered. During the execution, the number of active features varies so that the task is successively redundant and full-redundant. Figures 29, 30 and 31 present the behavior of the control laws (27), (29) and (30) with such task.

At the beginning of the servo, five features already belong to the confidence interval. The input signal is redundant but not full redundant. For all the executions, when the seventh feature is activated (around iteration 180), the input becomes full-redundant. Then, until the end, the active-feature number varies, but the input signal stays full-redundant. Since the features are incompatible, the global minimum $\mathbf{e} = \mathbf{0}$ is unattainable and the system converges to a local minimum. Most features are then outside the confidence interval they should reach, and some features stay at the border of the interval without being inactivated. The number of active features after convergence is 9.

Like in the second experiment, the three control laws present the same continuity properties when the input is not full redundant. In particular, a peak of acceleration appears each time a feature is activated in the three cases. This confirms that the three control laws may suffer of discontinuity problems when the input is not full redundant.

At iteration 180 the seventh feature is activated, and the system is then always full redundant. As expected, the control law (29) is then always continuous at feature activation or inactivation. On the opposite, control laws (27) and (30) are still discontinuous.

This experiment sums up the global behavior of the three control laws. When the added or removed feature is redundant with respect to the other input features, the proper way to build a varying-feature-set control law is (29) (which is, in that case, numerically equivalent to (28)). However, neither (28) nor (29) are able to ensure the continuity if the input is not properly redundant. In that case, none of the other studied control laws that have been presented in Section 2 is able to ensure the continuity.

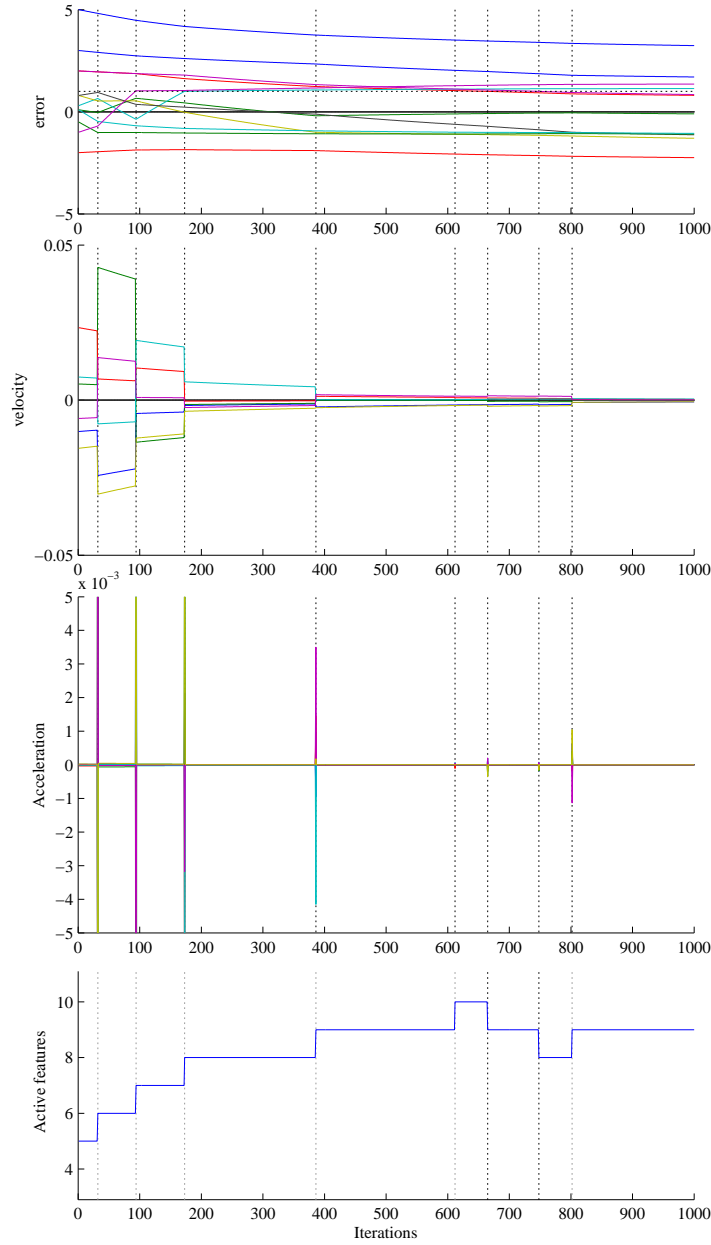


Figure 29: Exp. 4 ($\dim \mathbf{e} = 12$): the active input is not full redundant at the beginning of the servo, and becomes full-redundant when the 7th feature is activated. Control law (27), $\dot{\mathbf{q}} = -\lambda(\hat{\mathbf{H}}\mathbf{J})^+ \hat{\mathbf{H}}\mathbf{e}$, is used. The control law is always discontinuous at feature activation or inactivation.

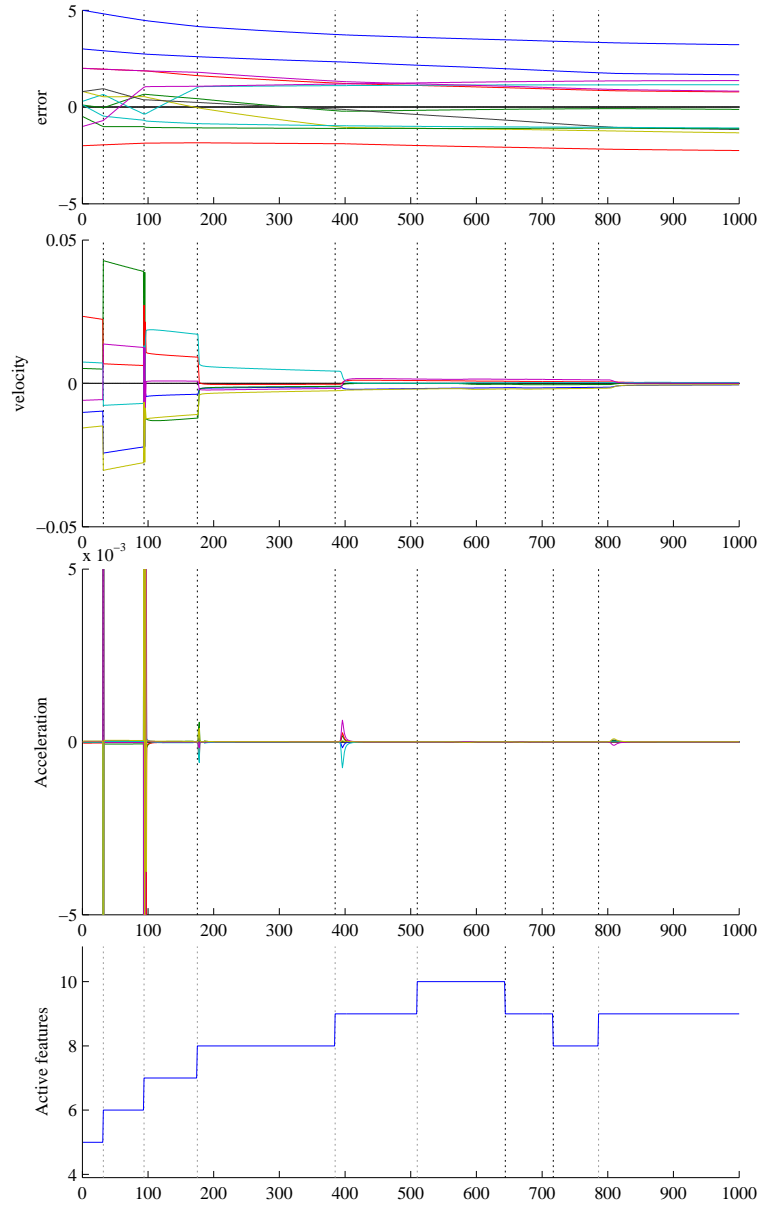


Figure 30: Exp. 4 ($\dim \mathbf{e} = 12$): control law (30), $\dot{\mathbf{q}} = -\lambda(\mathbf{H}\mathbf{J})^\dagger \mathbf{H}\mathbf{e}$, is used. The control law is discontinuous when the input is not full redundant (before iteration 180), and becomes continuous as soon as the active input is full redundant.

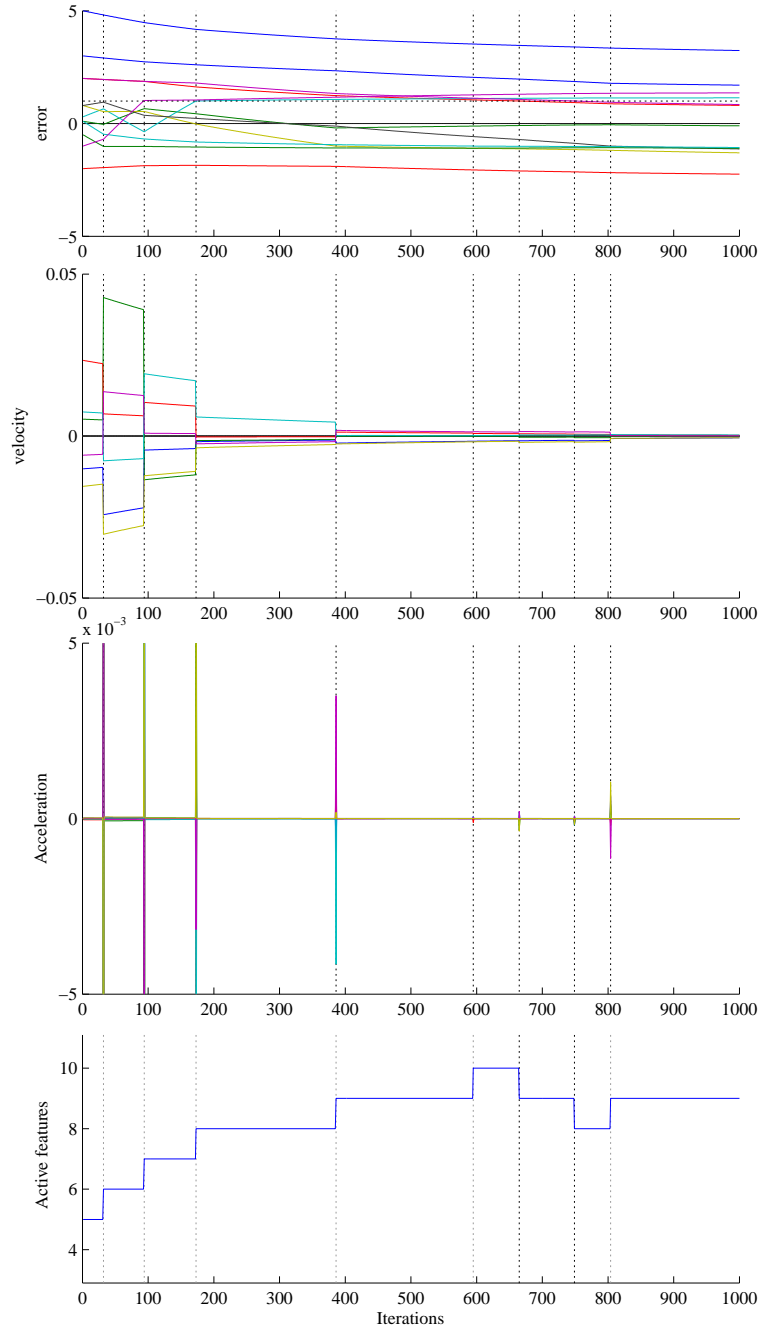


Figure 31: Exp. 4 ($\dim \mathbf{e} = 12$): control law (30), $\dot{\mathbf{q}} = -\lambda(\mathbf{H}\mathbf{J})^+ \hat{\mathbf{H}}\mathbf{e}$, is used. The control law is always discontinuous at feature activation of inactivation.

A.5 Fifth experiment: decoupled features

It has been seen that none of the control laws presented in Section 2 are able to ensure the continuity when the input is non-redundant. In this experiment, we study the discontinuity when the coupling between the inactivated feature and the other ones is null or very small.

The jacobian of the considered task can be decomposed as:

$$\mathbf{J} = \begin{bmatrix} \mathbf{J}_1 \\ \mathbf{J}_2^\perp + \chi \mathbf{J}_1 \end{bmatrix} \quad (75)$$

where \mathbf{J}_1 is any FRR matrix, \mathbf{J}_2^\perp is a one-row matrix belonging to the orthogonal of \mathbf{J}_1 and χ is any vector.

If the second part of the jacobian \mathbf{J} is reduced to \mathbf{J}_2^\perp (*i.e.* $\chi = \mathbf{0}$), then the feature corresponding to \mathbf{J}_2 is perfectly decoupled from the others. In this experiment, we have chosen $\|\chi\|$ very small, so that the decoupling is nearly perfect.

Two control laws are here compared, the one with a partial approximation, (30), and the one without any approximation, (28). They are recalled for the convenience of the reader:

$$\begin{array}{ll} \text{partial approximation} & \dot{\mathbf{q}} = -\lambda(\widehat{\mathbf{H}}\mathbf{J})^+ \mathbf{H}\mathbf{e} \\ \text{no approximation} & \dot{\mathbf{q}} = -\lambda(\mathbf{H}\mathbf{J})^+ \mathbf{H}\mathbf{e} \end{array}$$

Fig. 32 presents the evolution of the two control law norms when the feature corresponding to \mathbf{J}_2 is activated. As shown in Section 4.3, the obtained graph is piecewise constant when considering (28). A large discontinuity occurs near $h = 0$. When considering (30), a discontinuity also occurs at 0. However, it is much smaller. Furthermore, the obtained curve is then not constant as with the other control laws, but continuously evolves until the feature is fully activated.

This link between decoupling and continuity of (30) clearly appears on the following experiment presented on Fig 33. Control law (30) is used to consider a task where a decoupled feature is activated in a first time, while a non-coupled one is inactivated in a second time. When a decoupled feature is inactivated, there is no discontinuity in the control law. On the opposite, when the feature is coupled, a discontinuity clearly appears.

This discontinuity is directly correlated with the coupled part $\chi \mathbf{J}_1$. On Figure 34, the variation of the norm of the control law with respect to the variation of the norm of χ is plotted. The discontinuity increases with the coupling. In particular, when the coupling is null ($\chi = \mathbf{0}$), control law (30) is continuous.

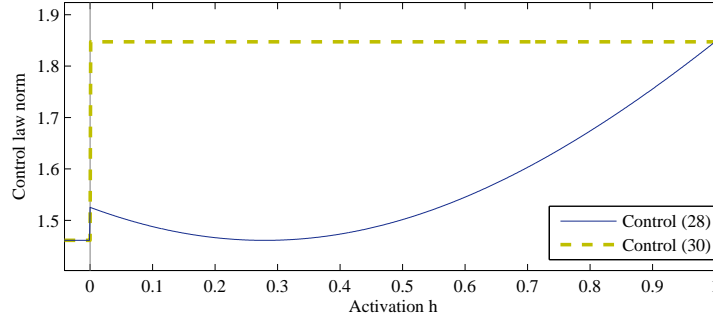


Figure 32: Exp. 5. Comparison between (28) and (30) when activating a quasi-decoupled feature. Control law (28) is clearly discontinuous (it is piecewise constant). On the opposite when using control law (30), the discontinuity is very small at $h = 0$, and the control law is then continuous until the feature is fully activated.

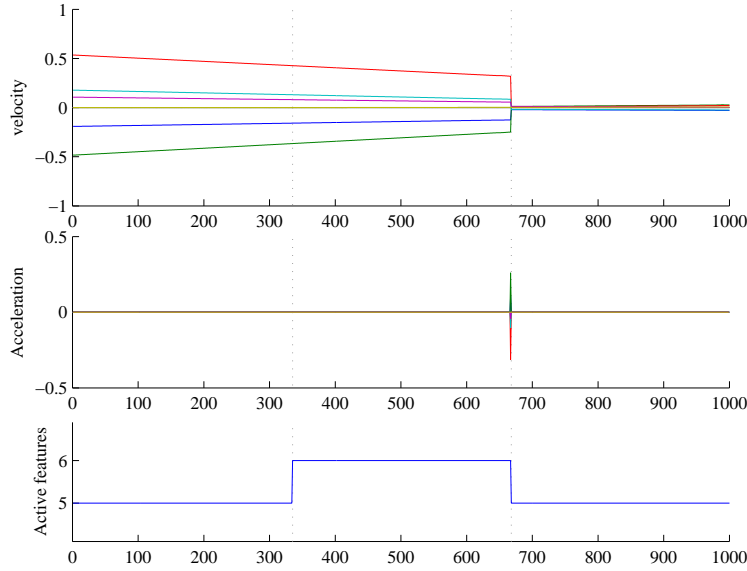


Figure 33: Exp. 5. Behavior of the control law (30) in presence of a decoupled feature. The task has six input signals and controls six DOF (it is full row rank). The last feature \mathbf{e}_6 is decoupled from the other ones. At Iteration 350, this feature leaves the qualitative area. Since it is decoupled, the control law remains continuous. At Iteration 680, another feature (the fifth one) enters the qualitative area. Since it is coupled to the others, the control law is discontinuous at this iteration.

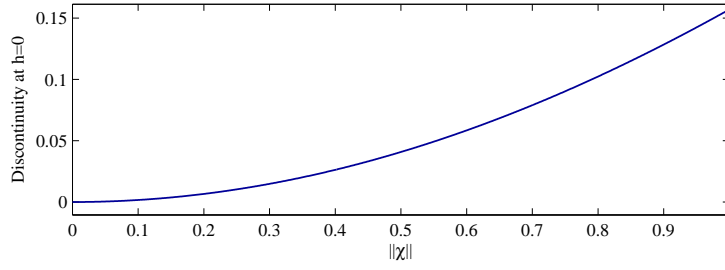


Figure 34: Exp. 5. Discontinuity in control law (30) with respect to the norm of χ . The discontinuity increases with the coupling between the features. In particular, when the feature is perfectly decoupled, control law (30) is constant.

B Continuous inverse versus varying feature sets

The study performed in the previous section is extended by considering the control law (66) based on the continuous inverse operator defined in Section 5, recalled here:

$$\dot{\mathbf{q}} = \mathbf{J}^{\oplus \mathbf{H}} \mathbf{e}_{\mathbf{q}}$$

Three experiments are presented to illustrate the behavior of this control law, along with some comparisons with the previous control laws (29) and (30). The same experimental protocol is used. The robotic system considered has six DOF, and, depending on the experiment, the task has two, six or twelve input signals. A qualitative convergence is required using (15). As soon as a feature error reaches the confidence interval $[0, 1]$, it is progressively inactivated. These experiments are simulations.

B.1 First experiment: $\dim \mathbf{e} = 2$

The task considered in this first experiment has two input features and is non-redundant. The experiment is summarized in Figures 35, 36 and 37 (for respectively the control law without approximation, with partial approximation, and with continuous inverse operator). As expected, when the second feature (the green one on the upper graph) gets activated (iteration 100) an acceleration peak is observed with the two first control laws, since the smoothing effect of the activation matrix is non effective in such situation. Fig. 37 shows that with the proposed continuous inverse operator, no acceleration peak occurs, and the velocity remains continuous even when the point is activated.

B.2 Second experiment: $\dim \mathbf{e} = 6$

The task is the same than in Section A.2. It has six features which correspond to a non-redundant input. The three control laws (no approximation, partial approximation, continuous inverse) are compared through executions presented on Figures 38, 39 and 40. At each

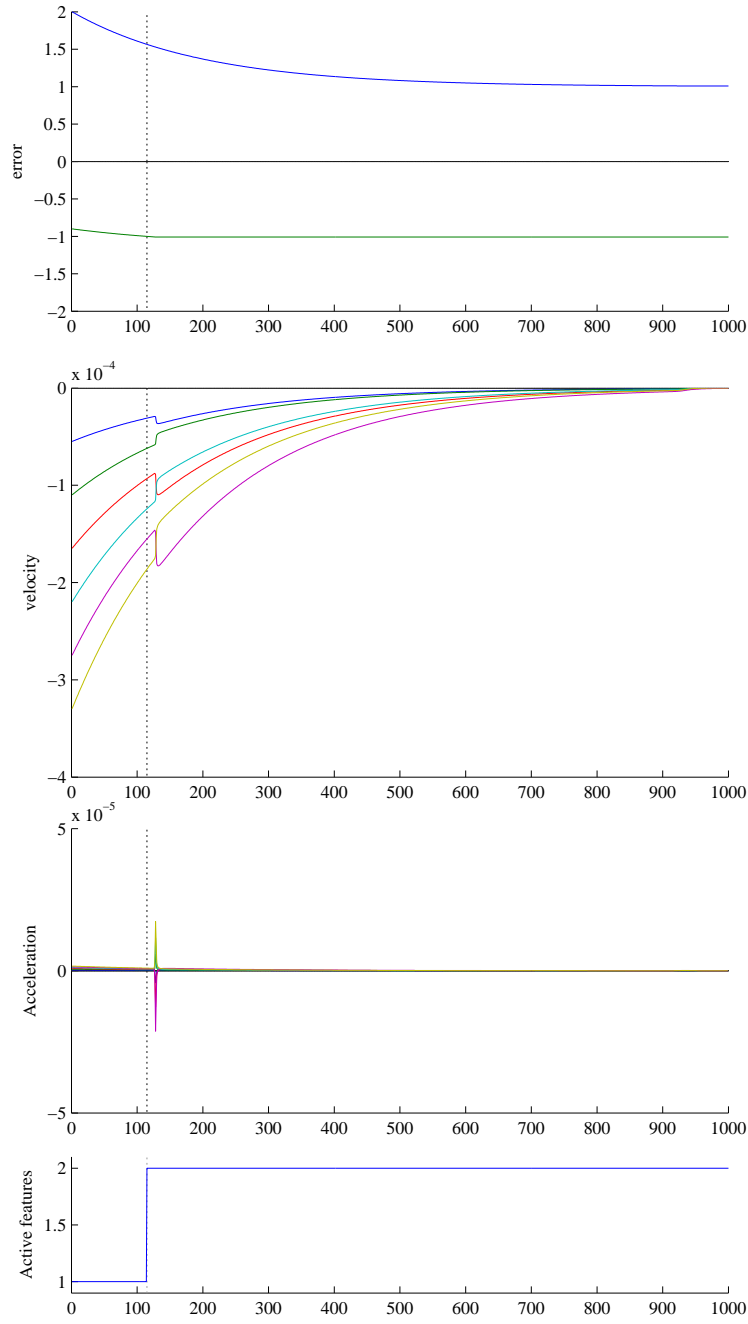


Figure 35: Exp. 1 ($\dim \mathbf{e} = 2$): the control law with no approximation (29) is used. The continuity is not ensured. Due to the use of the damped-least-square operator, the discontinuity is not at feature activation (at dash-line) but is delayed some iterations later.

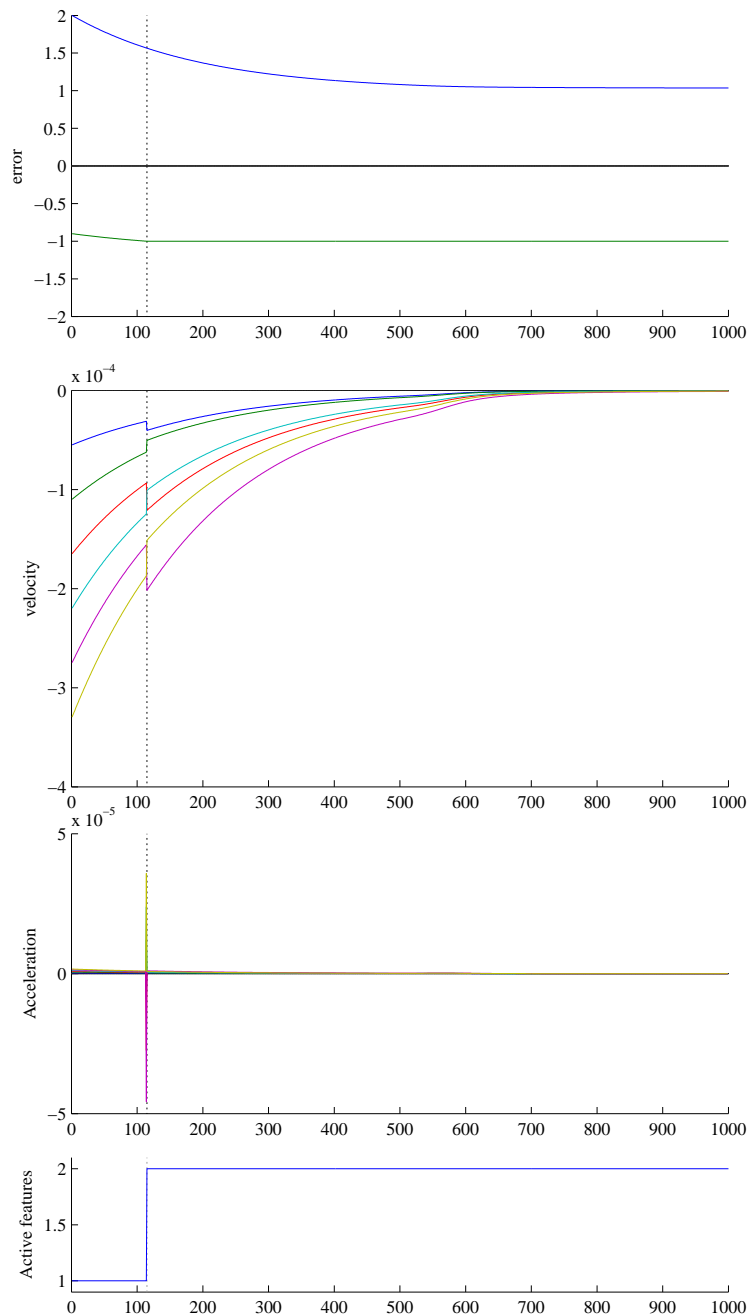


Figure 36: Exp. 1 ($\dim \mathbf{e} = 2$): the control law with a partial approximation (30) is used. The continuity is not ensured when the second feature gets activated.

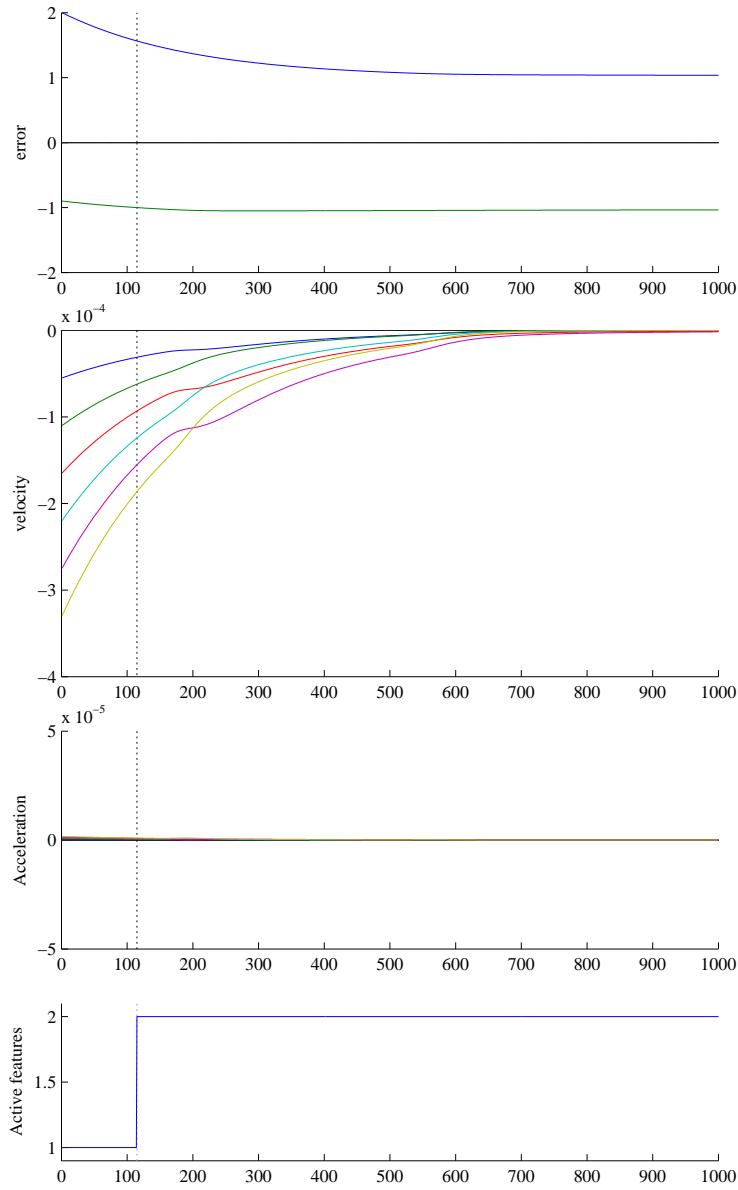


Figure 37: Exp. 1 ($\dim \mathbf{e} = 2$) : the control law with the continuous inverse operator (66) is used. The continuity is ensured everywhere. The velocities are smooth, and no peak of acceleration appears.

activation or inactivation (iterations 30 and 150), control laws (29) or (30) are discontinuous. With the continuous inverse operator (66), these discontinuities do not appear anymore. Naturally, some accelerations occurs when a feature is added/removed. Nevertheless these accelerations do not correspond to peaks as observed with the two previous control laws.

B.3 Third experiment: $\dim \mathbf{e} = 12$

This task is exactly the one used in Section A.4. At the beginning of the servo, five features are activated. The task is non-redundant. The task becomes then redundant around iteration 100 when the fifth feature is activated. The experiment is summarized in Fig. 41, and 42. Control law (30) is not given since it has been proved to be not relevant when the task is full redundant.

As long as the active input is non-redundant, each activation or inactivation produces a discontinuity in the control law (29). By using (66) the continuity is ensured. As soon as the input is redundant, (29) becomes continuous. The two control laws (29) and (66) produce then a similar behavior, as it was required by the definition 5.1 of a general continuous inverse operator.

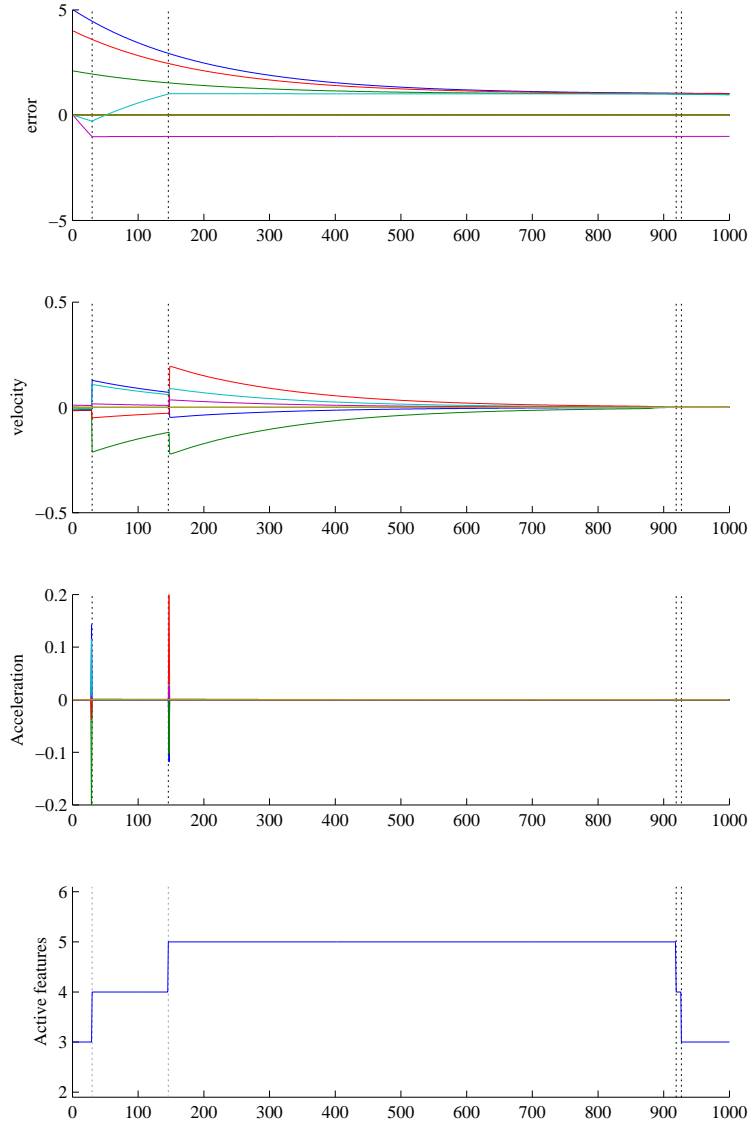


Figure 38: Exp. 2 ($\dim \mathbf{e} = 6$): the control law (29) with no approximation is used. The continuity is not ensured. Acceleration peaks are observed at iterations 30 and 150, when two of the initially non active features are getting active.

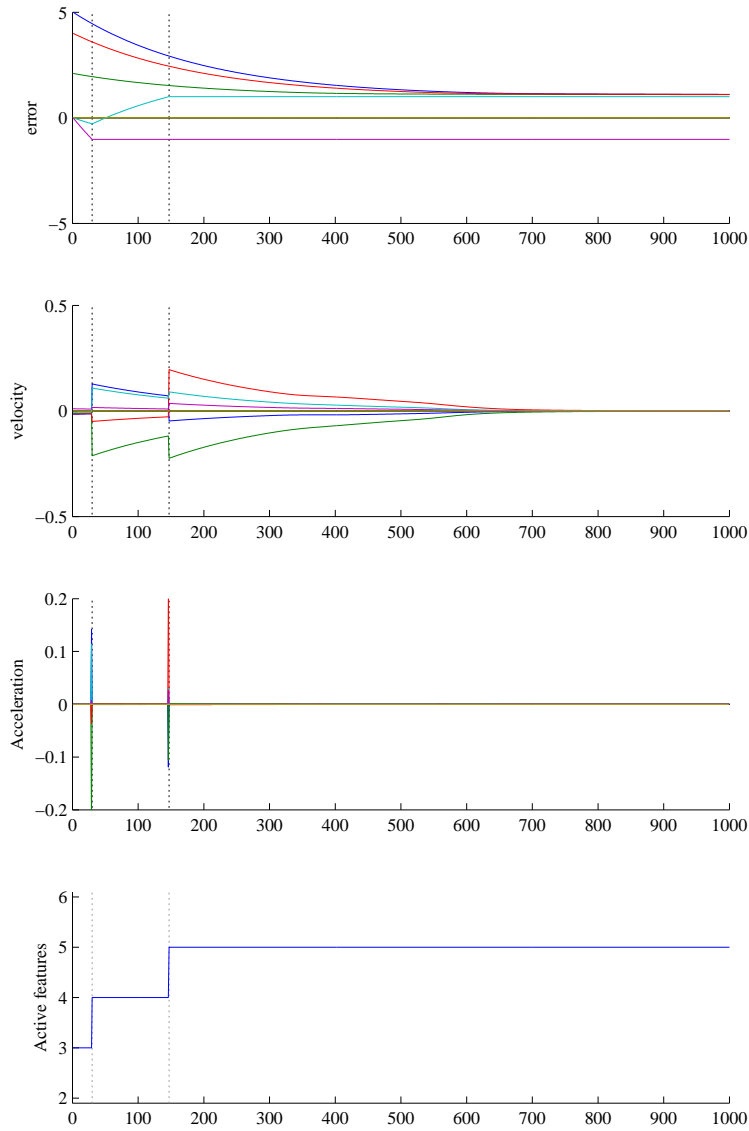


Figure 39: Exp. 2 ($\dim \mathbf{e} = 6$): the control law (30) with a partial approximation is used. As long as the input features are not full redundant, the behavior obtained is the same than with the control law without approximation.

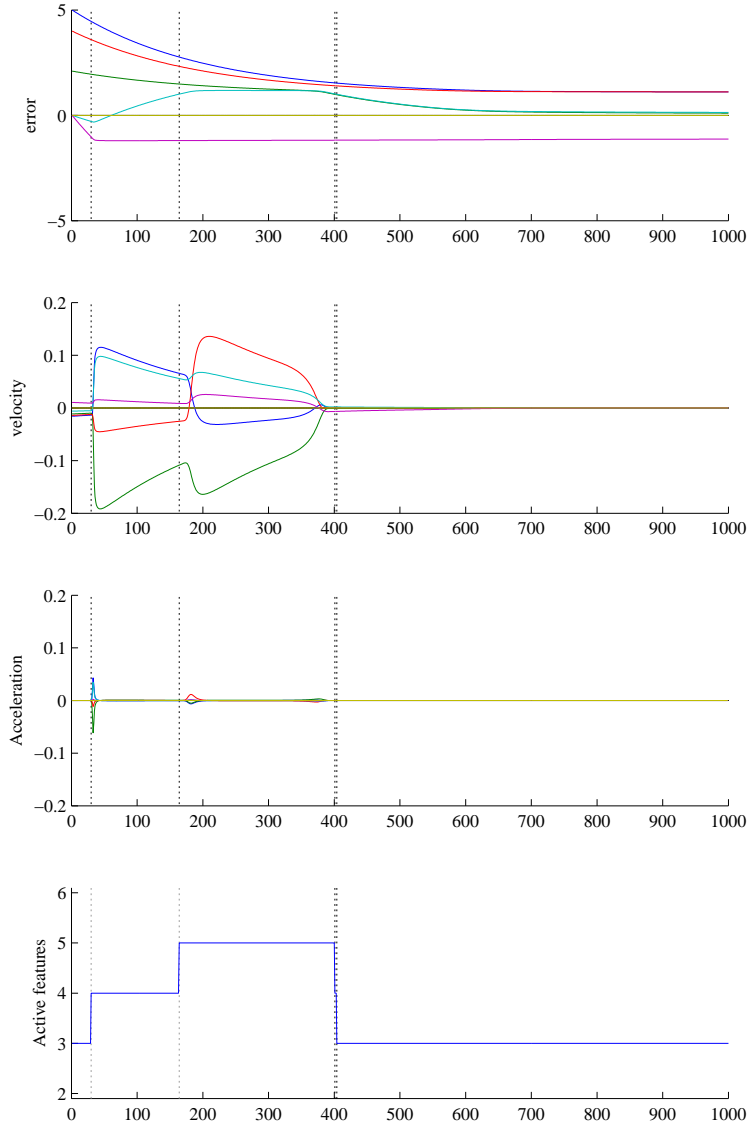


Figure 40: Exp. 2 ($\dim \mathbf{e} = 6$). The control law (66) using a continuous inverse operator is here illustrated. The continuity is ensured everywhere. The velocities are smooth, and no peak of acceleration appears.

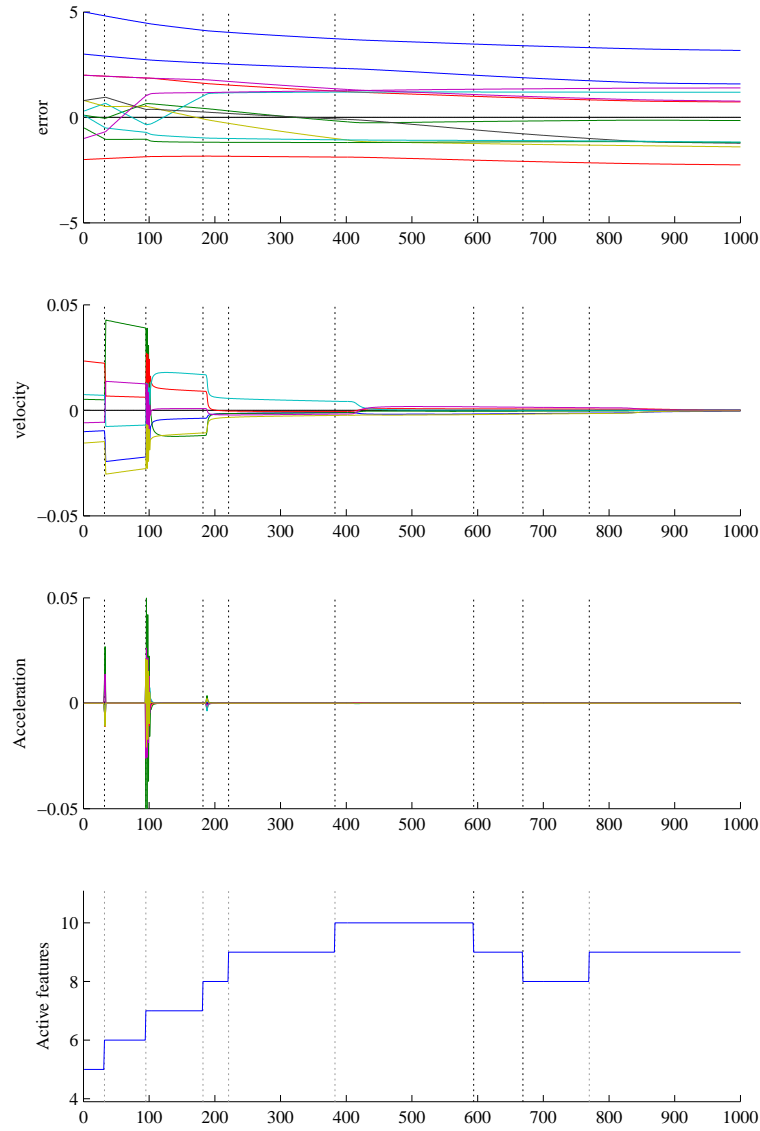


Figure 41: Exp. 3 ($\dim \mathbf{e} = 12$) : the control law (29) without approximation is used. The continuity is not ensured everywhere. Some peaks of acceleration appear when the active feature input is non redundant (iterations 30 and 100).

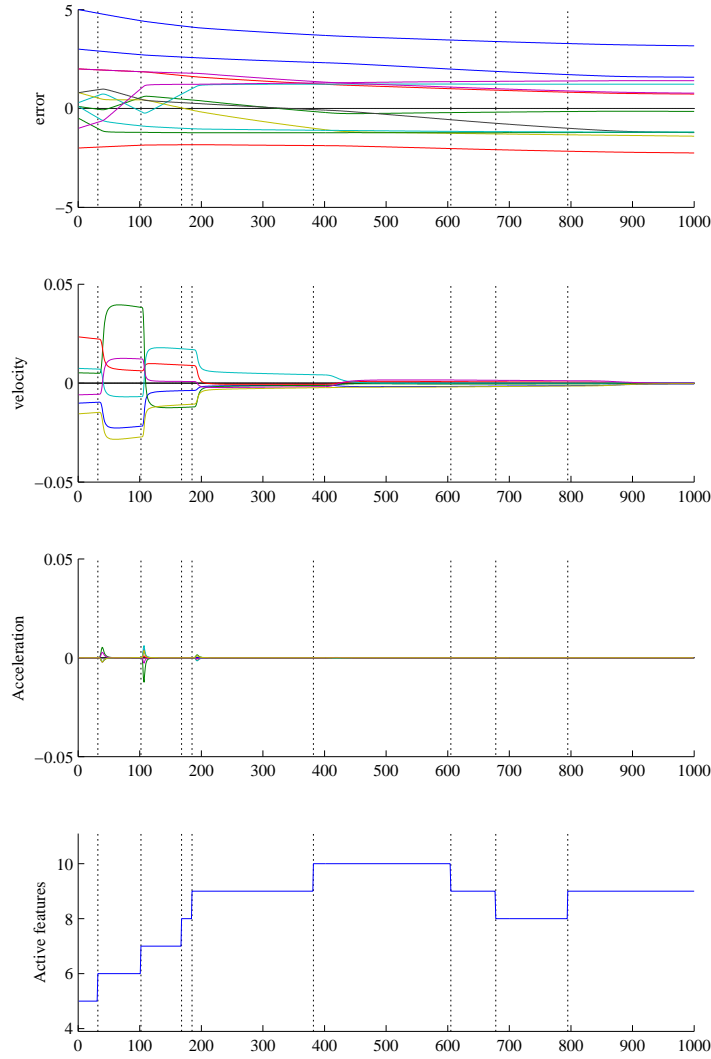


Figure 42: Exp. 3 ($\dim \mathbf{e} = 6$): the control law (66) with the continuous inverse operator is used. The velocity computed is much more smooth than in the previous experiment. The continuity is ensured everywhere. No peak of acceleration appears. In particular, the continuity is ensured when the input is non redundant. When the active features are redundant, the robot behavior is similar to the one obtained using the previous control law (29).

References

- [1] P. Baerlocher. *Inverse kinematics techniques for the interactive posture control of articulated figures*. PhD thesis, EPFL, 2001.
- [2] A. Ben-Israel and T.N.E Greville. *Generalized inverses: theory and applications*. Wiley, New York, 1980.
- [3] T.-F. Chang and R.-V. Dubey. A weighted least-norm solution based scheme for avoiding joints limits for redundant manipulators. *IEEE Trans. on Robotics and Automation*, 11(2):286 – 292, April 1995.
- [4] F. Chaumette. Potential problems of stability and convergence in image-based and position-based visual servoing. In D. Kriegman, G. Hager, and A.S. Morse, editors, *The Confluence of Vision and Control*, pages 66–78. LNCIS Series, No 237, Springer-Verlag, 1998.
- [5] C.C. Cheah and D.Q. Wang. Region reaching control of robots: Theory and experiments. In *IEEE Int. Conf. on Robotics and Automation (ICRA '05)*, pages 986 – 992, Barcelona, Spain, May 2005.
- [6] A.I. Comport, E. Marchand, and F. Chaumette. Statistically robust 2d visual servoing. *IEEE Trans. on Robotics*, 22(2):415 – 421, April 2006.
- [7] P.I. Corke and S.A. Hutchinson. A new partitioned approach to image-based visual servo control. *IEEE Transactions on Robotics and Automation*, 17(4):507 – 515, August 2001.
- [8] A.S. Deo and I.D. Walker. Robot subtask performance with singularity robustness using optimal damped least squares. In *IEEE Int. Conf. on Robotics and Automation (ICRA '92)*, pages 434 – 441, Nice, France, May 1992.
- [9] K.L. Doty, C. Melchiorri, and C. Bonivento. A theory of generalized inverses applied to robotics. *Int. J. Robotics Research*, 12(1):1 – 19, 1993.
- [10] B. Espiau, F. Chaumette, and P. Rives. A new approach to visual servoing in robotics. *IEEE Trans. on Robotics and Automation*, 8(3):313–326, June 1992.
- [11] N. Garcia-Aracil, E. Malis, R. Aracil-Santonja, and C. Perez-Vidal. Continuous visual servoing despite the changes of visibility in image features. *IEEE Transaction on Robotics*, 21(6), 2005.
- [12] S. Hutchinson, G. Hager, and P. Corke. A tutorial on visual servo control. *IEEE Trans. on Robotics and Automation*, 12(5):651–670, October 1996.
- [13] L. Kelmar and P. K. Khosla. Automatic generation of forward and inverse kinematics of a reconfigurable modular manipulator system. *Journal of Robotic Systems*, pages 599 – 620, August 1990.

- [14] E. Malis. Improving vision-based control using efficient second-order minimization techniques. In *IEEE Int. Conf. on Robotics and Automation (ICRA '04)*, New Orleans, USA, April 2004.
- [15] E. Malis. Visual servoing invariant to changes in camera intrinsic parameters. *IEEE Trans. on Robotics and Automation*, 20(1):72–81, February 2004.
- [16] E. Malis, F. Chaumette, and S. Boudet. 2 1/2 D visual servoing. *IEEE Trans. on Robotics and Automation*, 15(2):238–250, April 1999.
- [17] E. Marchand and G. Hager. Dynamic sensor planning in visual servoing. In *IEEE/RSJ Int. Conf. on Intelligent Robots and Systems (IROS'98)*, volume 3, pages 1988–1993, Leuven, Belgium, May 1998.
- [18] Y. Nakamura and H. Hanafusa. Inverse kinematics solutions with singularity robustness for robot manipulator control. *Trans. ASME Journal of Dynamic System, Measures and Control*, 108:163 – 171, 1986.
- [19] B. Nelson and P.K. Khosla. Strategies for increasing the tracking region of an eye-in-hand system by singularity and joint limits avoidance. *Int. Journal of Robotics Research*, 14(3):255 – 269, June 1995.
- [20] A. Remazeilles, N. Mansard, and F. Chaumette. Qualitative visual servoing: application to the visibility constraint. In *IEEE/RSJ Int. Conf. on Intelligent Robots and Systems (IROS'06)*, 2006.
- [21] C. Samson, M. Le Borgne, and B. Espiau. *Robot Control: the Task Function Approach*. Clarendon Press, Oxford, United Kingdom, 1991.
- [22] B. Siciliano and J-J. Slotine. A general framework for managing multiple tasks in highly redundant robotic systems. In *ICAR'91*, pages 1211 – 1216, 1991.
- [23] D.E. Whitney. The mathematics of coordinated control of prosthetic arms and manipulators. *Trans. ASME Journal of Dynamic System, Measures and Control*, 94:303 – 309, 1972.